User's Manual

32

V850E2/Sx4-H

Hardware User's Manual

RENESAS MCUs
V850E2/Sx4-H microcontrollers [Preliminary]

Renesas Electronics
www.renesas.com

R01UH0107EJ0002, Rev. 0.02
September 30, 2011

Notes for CMOS devices

(1) Voltage application wafeform at input pin:

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between VIL (MAX) and VIH (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between VIL (MAX) and VIH (MIN).

(2) Handling of unused input pins:

Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to VDD or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

(3) Precaution against ESD:

A strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

(4) Status before initialization:

Power-on does not necessarily define the initial status of a MOS device. Immediately after the power source is turned ON, devices with reset functions have not yet been initialized. Hence, power-on does not guarantee output pin levels, I/O settings or contents of registers. A device is not initialized until the reset signal is received. A reset operation must be executed immediately after power-on for devices with reset functions.

(5) Power ON/OFF sequence:

In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current. The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

(6) Input of signal during power off state:

Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

How to Use This Manual

Readers

This manual is intended for users who wish to understand the functions of the V850E2/Sx4-H and design application systems using the following V850E2/Sx4-H microcontrollers:

- V850E2/SG4-H: μPD70F4013, 70F4014
- V850E2/SJ4-H: μPD70F4015, 70F4016
- V850E2/SK4-H: μPD70F4017, 70F4018

Purpose

This manual is intended to give users an understanding of the hardware functions of the V850E2/Sx4-H shown in the *Organization* below.

Organization

This manual is divided into two parts: Hardware (this manual) and Architecture (V850E2M Architecture User's Manual).

Hardware	Architecture
Pin functions	Data types
CPU function	Register set
On-chip peripheral functions	Instruction format and instruction set
Flash memory programming	Interrupts and exceptions
	Pipeline operation

How to read this manual

It is assumed that the readers of this manual have general knowledge in the fields of electrical engineering, logic circuits, and microcontrollers.

To understand the overall functions of the V850E2/Sx4-H.

Read this manual according to the Contents.

To understand the details of an instruction function

See *V850E2M Architecture User's Manual* available separately.

The mark <R> shows major revised points. The revised points can be easily searched by copying an "<R>" in the PDF file and specifying it in the "Find what:" field.

Table of Contents

Chapter 1 Introduction

The V850E2/Sx4-H is a single-chip microcontroller from Renesas Electronics intended for use in automotive applications.

1.1 Overview

The V850E2/Sx4-H is a 32-bit single-chip microcontroller for embedded control applications and incorporates a V850E2M CPU that provides enhanced computing power with a compact code size comparable to that of a 16-bit CISC CPU.

The V850E2/Sx4-H provides a wide range of on-chip peripherals such as serial interfaces, timers and counters, and controllers for various features including CAN, MOST® (MediaLB®), and Ethernet bus networks.

The V850E2/Sx4-H also provides several power-saving modes, enabling developers to create power-efficient systems that operate under a range of power supply conditions.

These features make the V850E2/Sx4-H perfect for use in car audio and many other car multimedia applications.

1.2 V850E2/Sx4-H Product Overview

The V850E2/Sx4-H lineup consists of theV850E2/SG4-H in a 100-pin QFP, the V850E2/SJ4-H in a 144-pin QFP, and the V850E2/SK4-H in a 176-pin QFP.

An overview of each product is shown below.

(1) V850E2/SG4-H product overview

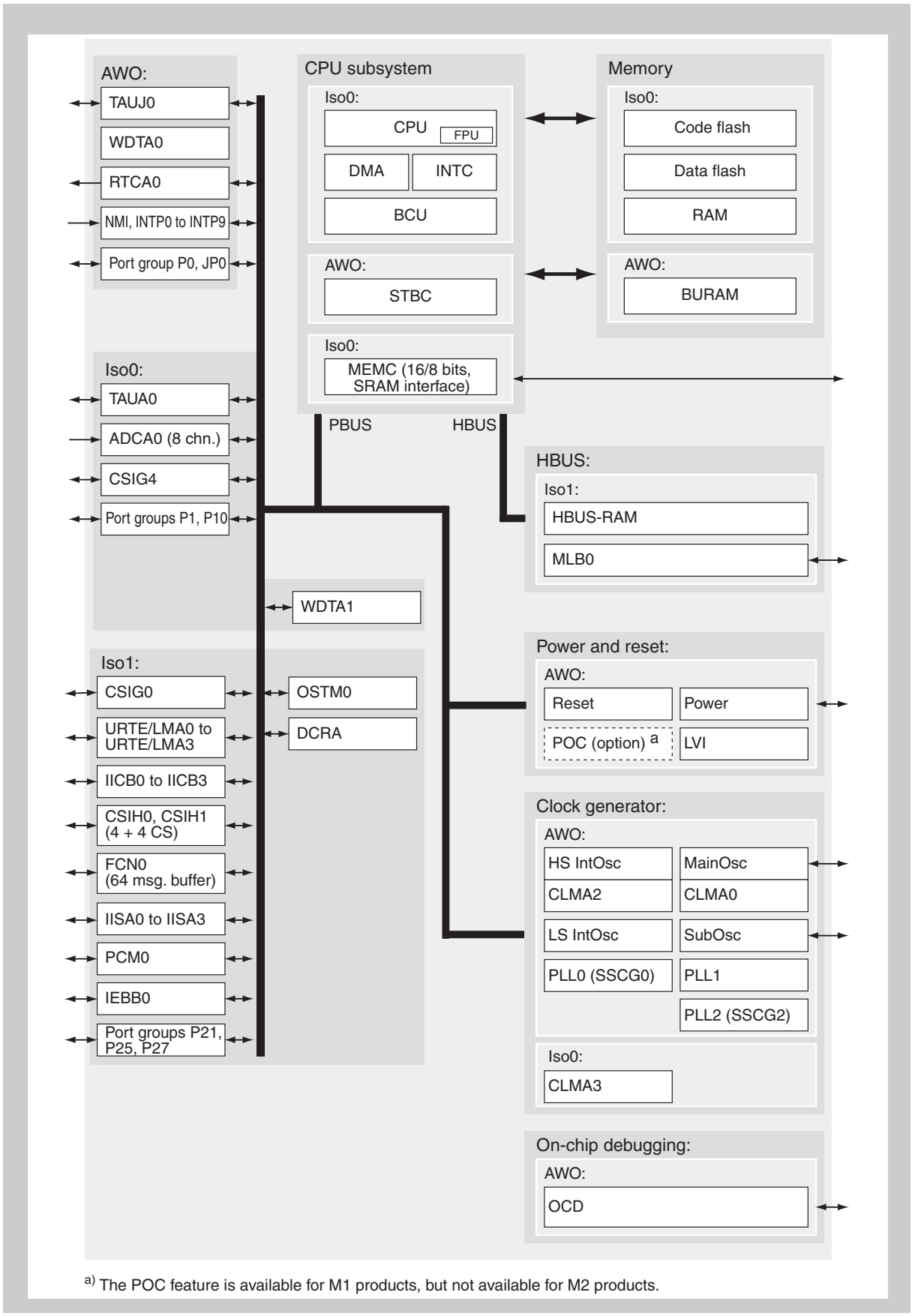
Table 1-1 V850E2/SG4-H products (1/2)

Commercial name			SG4-H-1M	SG4-H-1.5M
Part number			μPD70F4013	μPD70F4014
Internal memory	Code flash		1 MB	1.5 MB
	Data flash		32 KB	
	CPU RAM		96 KB	128 KB
	HBUS-RAM		32 KB	
	Backup RAM		32 KB	
External memory interface (MEMC)			Multiplexed SRAM interface (8/16 bits)	
CPU	CPU system		V850E2M	
	FPU		Yes	
	CPU frequency		160 MHz (MAX.)	
	System protection feature (SPF)	MPU	Yes	
		SRP	Yes	
		TSU	Yes	
		PPU	Yes	
	Instruction cache		8 KB/2-way associative (4 KB/way)	
DMA			16 channels	
Operating clock	Main clock oscillator (MainOsc)		4 MHz to 20 MHz	
	Low-speed internal oscillator (low-speed IntOsc)		240 kHz (TYP.)	
	High-speed internal oscillator (high-speed IntOsc)		8 MHz (TYP.)	
	Subclock oscillator (SubOsc)		32.768 kHz (TYP.)	
	PLL0 (SSCG0)		160 MHz (MAX.)	
	PLL1		120 MHz (MAX.)	
	PLL2 (SSCG2)		120 MHz (MAX.)	
<R> I/O ports			58	
<R> A/D converter A (ADCA)			8 channels, 10 bits	
Timers	Timer array unit A (TAUA), 16 bits		1 unit x 16 channels	
	Timer array unit B (TAUB), 16 bits		–	
	Timer array unit J (TAUJ), 32 bits		1 unit x 4 channels	
	Real-time clock (RTCA) calibration		1 unit	
	Window watchdog timer (WDTA)		2 channels	
	OS timer (OSTM)		1 channel	
	Encoder timer (ENCA)		–	

Table 1-1 V850E2/SG4-H products (2/2)

Commercial name			SG4-H-1M	SG4-H-1.5M
Part number			μPD70F4013	μPD70F4014
<R> Serial interfaces	CAN (FCN)		1 channel (64-message buffer)	
	UART with LIN master controller LMA (URTE)		4 channels	
	CSI (CSIG)		2 channels	
	CSI with FIFO (CSIH)		2 channels	
	I^2C (IICB)		4 channels	
	I^2S (IISA)		4 channels	
	PCM interface (PCM)		1 channel	
	Media local bus (MLB)		1 channel	
	IEBus controller (IEBB)		1 channel	
Other interfaces	Ethernet controller (ETHA)		–	
Interrupts	Maskable	External	10	
		Internal	144	
	Non-maskable (NMI)	External	1	
		Internal	2 (WDTA)	
Other features	Power-on-clear circuit (POC)		M1 products: Yes, M2 products: No	
	Clock monitor (CLMA)		3 channels (Can monitor main clock, high-speed internal oscillator, and PLL0)	
	Data CRC (DCRA)		1 channel	
	Key interrupt (KR)		–	
	On-chip debug unit (OCD)		Yes	
Power supply monitoring	Low-voltage indicator (LVI)		Yes	
	DEEPSTOP mode display output (WAKE)		Yes	
Power supply	Supplied internally		1.1 V to 1.3 V[a]	
	Supplied via I/O		3.0 V to 3.6 V[a]	
Operating temperature			(A) products: –40°C to +85°C, (A9) products: –40°C to +105°C	
Package			100-pin QFP	

a) For details, see the Data Sheet.



a) The POC feature is available for M1 products, but not available for M2 products.

<R> **Figure 1-1 Block diagram of V850E2/SG4-H**

(2) V850E2/SJ4-H product overview

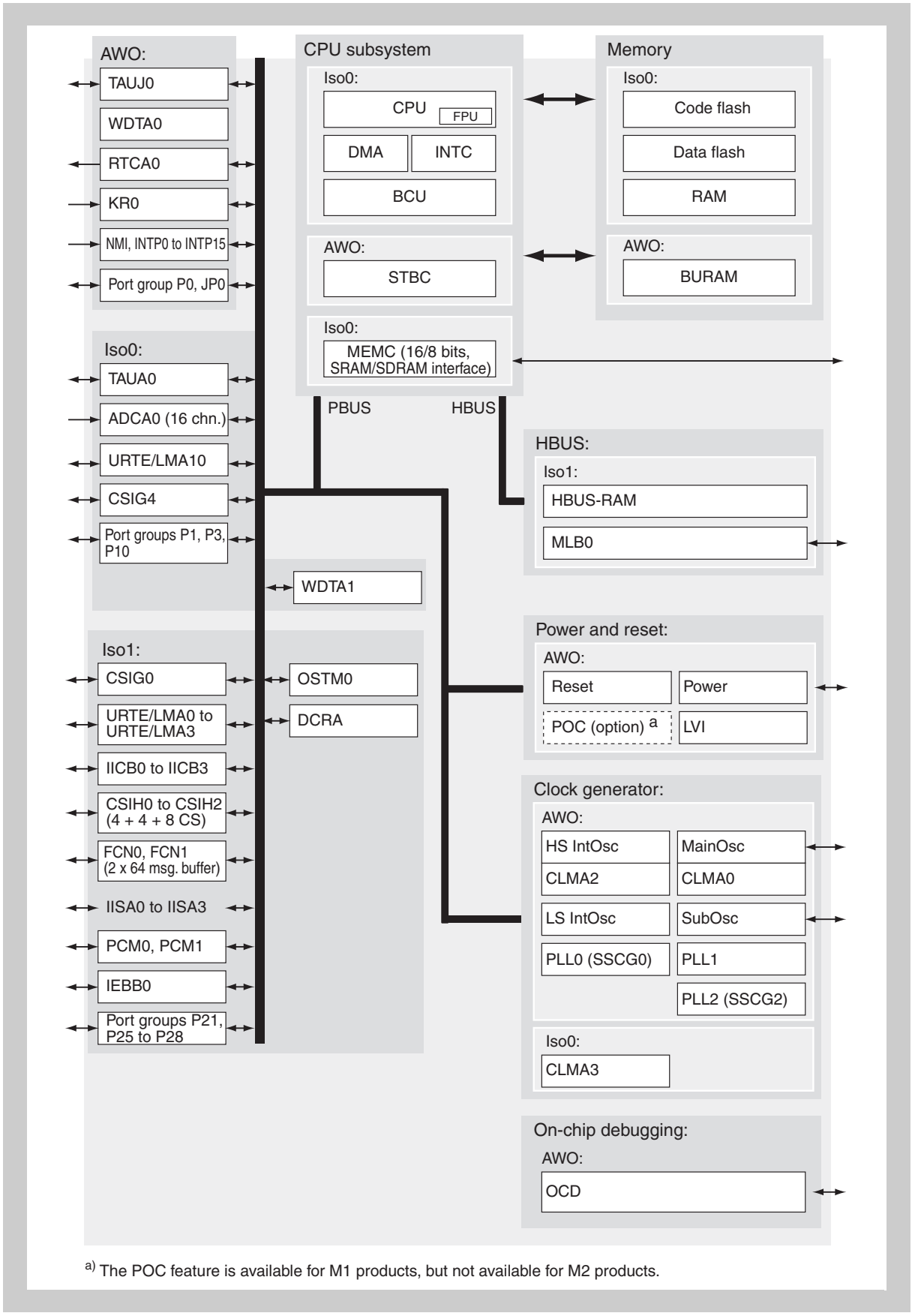
Table 1-2 V850E2/SJ4-H products (1/2)

	Commercial name			SJ4-H-1M	SJ4-H-1.5M
	Part number			μPD70F4015	μPD70F4016
	Internal memory	Code flash		1 MB	1.5 MB
		Data flash		32 KB	
		CPU RAM		96 KB	128 KB
		HBUS-RAM		32 KB	
		Backup RAM		32 KB	
	External memory interface (MEMC)			SDRAM interface Multiplexed/separate SRAM interface (8/16 bits)	
	CPU	CPU system		V850E2M	
		FPU		Yes	
		CPU frequency		160 MHz (MAX.)	
		System protection feature (SPF)	MPU	Yes	
			SRP	Yes	
			TSU	Yes	
			PPU	Yes	
		Instruction cache		8 KB/2-way associative (4 KB/way)	
	DMA			16 channels	
	Operating clock	Main clock oscillator (MainOsc)		4 MHz to 20 MHz	
		Low-speed internal oscillator (low-speed IntOsc)		240 kHz (TYP.)	
		High-speed internal oscillator (high-speed IntOsc)		8 MHz (TYP.)	
		Subclock oscillator (SubOsc)		32.768 kHz (TYP.)	
		PLL0 (SSCG0)		160 MHz (MAX.)	
		PLL1		120 MHz (MAX.)	
		PLL2 (SSCG2)		120 MHz (MAX.)	
<R>	I/O ports			100	
<R>	A/D converter A (ADCA)			16 channels, 10 bits	
	Timers	Timer array unit A (TAUA), 16 bits		1 unit x 16 channels	
		Timer array unit B (TAUB), 16 bits		–	
		Timer array unit J (TAUJ), 32 bits		1 unit x 4 channels	
		Real-time clock (RTCA) calibration		1 unit	
		Window watchdog timer (WDTA)		2 channels	
		OS timer (OSTM)		1 channel	
		Encoder timer (ENCA)		–	

Table 1-2 V850E2/SJ4-H products (2/2)

Commercial name			SJ4-H-1M	SJ4-H-1.5M
Part number			μPD70F4015	μPD70F4016
Serial interfaces	CAN (FCN)		2 channels (64-message buffer)	
	UART with LIN master controller LMA (URTE)		5 channels	
	CSI (CSIG)		2 channels	
	CSI with FIFO (CSIH)		3 channels	
	I^2C (IICB)		4 channels	
	I^2S (IISA)		64 channels	
	PCM interface (PCM)		62 channels	
	Media local bus (MLB)		1 channel	
	IEBus controller (IEBB)		1 channel	
Other interfaces	Ethernet controller (ETHA)		–	
Interrupts	Maskable	External	16	
		Internal	161	
	Non-maskable (NMI)	External	1	
		Internal	2 (WDTA)	
Other features	Power-on-clear circuit (POC)		M1 products: Yes, M2 products: No	
	Clock monitor (CLMA)		3 channels (Can monitor main clock, high-speed internal oscillator, and PLL0)	
	Data CRC (DCRA)		1 channel	
	Key interrupt (KR)		8 channels	
	On-chip debug unit (OCD)		Yes	
Power supply monitoring	Low-voltage indicator (LVI)		Yes	
	DEEPSTOP mode display output (WAKE)		Yes	
Power supply	Supplied internally		1.1 V to 1.3 V[a]	
	Supplied via I/O		3.0 V to 3.6 V[a]	
Operating temperature			(A) products: –40°C to +85°C, (A9) products: –40°C to +105°C	
Package			144-pin QFP	

a) For details, see the Data Sheet.



a) The POC feature is available for M1 products, but not available for M2 products.

Figure 1-2 Block diagram of V850E2/SJ4-H

(3) V850E2/SK4-H product overview

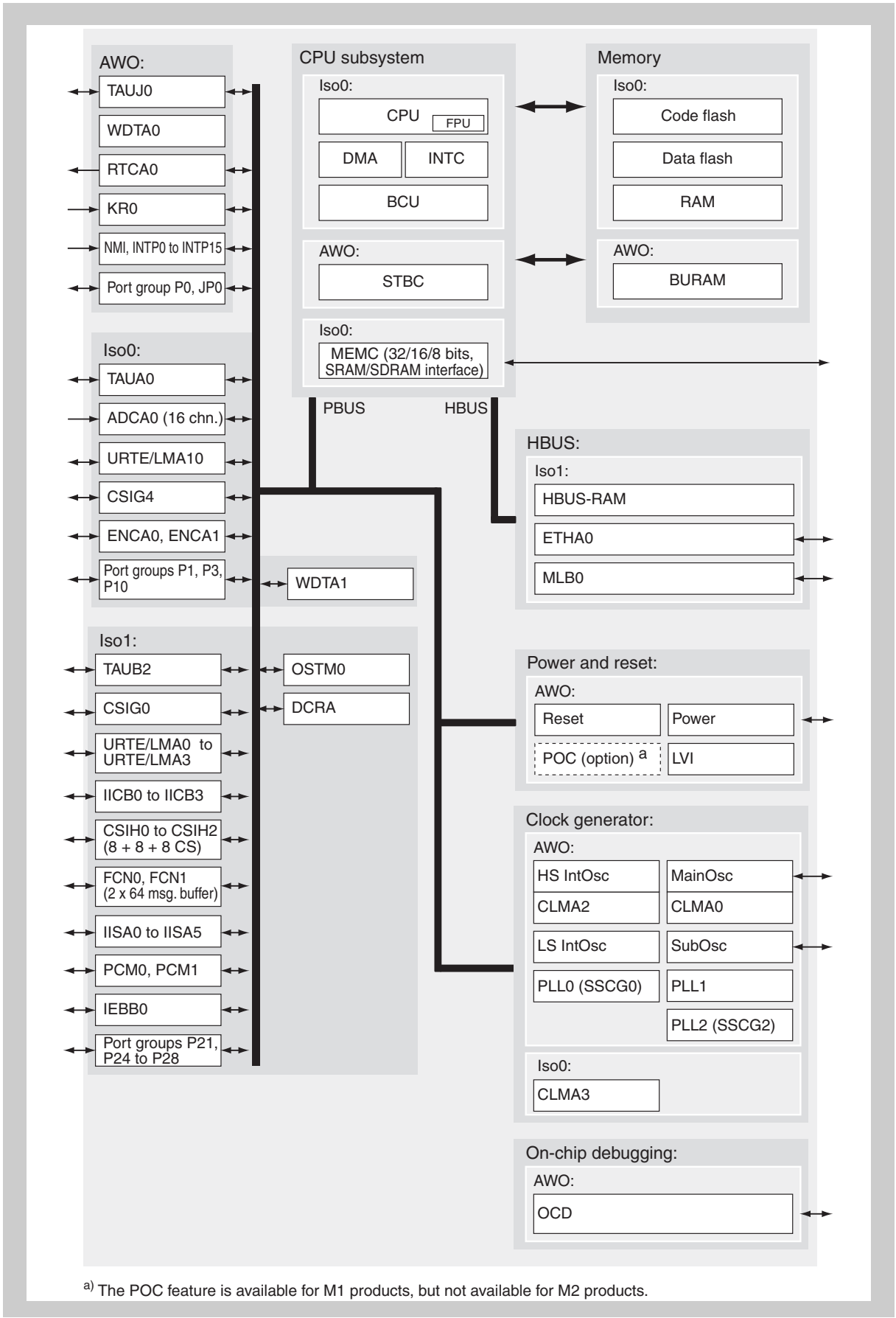
Table 1-3 V850E2/SK4-H products (1/2)

Commercial name			SK4-H-1.5M	SK4-H-2M
Part number			μPD70F4017	μPD70F4018
Internal memory	Code flash		1.5 MB	2 MB
	Data flash		32 KB	
	CPU RAM		128 KB	192 KB
	HBUS-RAM		32 KB	
	Backup RAM		32 KB	
External memory interface (MEMC)			SDRAM interface Multiplexed/separate SRAM interface (8/16/32 bits)	
CPU	CPU system		V850E2M	
	FPU		Yes	
	CPU frequency		160 MHz (MAX.)	
	System protection feature (SPF)	MPU	Yes	
		SRP	Yes	
		TSU	Yes	
		PPU	Yes	
	Instruction cache		8 KB/2-way associative (4 KB/way)	
DMA			16 channels	
Operating clock	Main clock oscillator (MainOsc)		4 MHz to 20 MHz	
	Low-speed internal oscillator (low-speed IntOsc)		240 kHz (TYP.)	
	High-speed internal oscillator (high-speed IntOsc)		8 MHz (TYP.)	
	Subclock oscillator (SubOsc)		32.768 kHz (TYP.)	
	PLL0 (SSCG0)		160 MHz (MAX.)	
	PLL1		120 MHz (MAX.)	
	PLL2 (SSCG2)		120 MHz (MAX.)	
I/O ports			127	
A/D converter A (ADCA)			16 channels, 10 bits	
Timers	Timer array unit A (TAUA), 16 bits		1 unit x 16 channels	
	Timer array unit B (TAUB), 16 bits		1 unit x 16 channels	
	Timer array unit J (TAUJ), 32 bits		1 unit x 4 channels	
	Real-time clock (RTCA) calibration		1 unit	
	Window watchdog timer (WDTA)		2 channels	
	OS timer (OSTM)		1 channel	
	Encoder timer (ENCA)		2 channels	

Table 1-3 V850E2/SK4-H products (2/2)

Commercial name			SK4-H-1.5M	SK4-H-2M
Part number			μPD70F4017	μPD70F4018
Serial interfaces	CAN (FCN)		2 channels (64-message buffer)	
	UART with LIN master controller LMA (URTE)		5 channels	
	CSI (CSIG)		2 channels	
	CSI with FIFO (CSIH)		3 channels	
	I^2C (IICB)		4 channels	
	I^2S (IISA)		6 channels	
	PCM interface (PCM)		2 channels	
	Media local bus (MLB)		1 channel	
	IEBus controller (IEBB)		1 channel	
Other interfaces	Ethernet controller (ETHA)		1 unit (supports MII interface)	
Interrupts	Maskable	External	16	
		Internal	208	
	Non-maskable (NMI)	External	1	
		Internal	2 (WDTA)	
Other features	Power-on-clear circuit (POC)		M1 products: Yes, M2 products: No	
	Clock monitor (CLMA)		3 channels (Can monitor main clock, high-speed internal oscillator, and PLL0)	
	Data CRC (DCRA)		1 channel	
	Key interrupt (KR)		8 channels	
	On-chip debug unit (OCD)		Yes	
Power supply monitoring	Low-voltage indicator (LVI)		Yes	
	DEEPSTOP mode display output (WAKE)		Yes	
Power supply	Supplied internally		1.1 V to 1.3 V[a]	
	Supplied via I/O		3.0 V to 3.6 V[a]	
Operating temperature			(A) products: –40°C to +85°C, (A9) products: –40°C to +105°C	
Package			176-pin QFP	

a) For details, see the Data Sheet.



a) The POC feature is available for M1 products, but not available for M2 products.

Figure 1-3 Block diagram of V850E2/SK4-H

1.3 Related Documents

Table 1-4 Related Documents

Document number	Title
U19949E	V850E2M Architecture User's Manual

1.4 Ordering Information

<R> Table 1-5 V850E2/Sx4-H ordering information

Commercial name	Device name	Renesas ordering code	Grade	Note
SG4-H-1M	μ PD70F4013	μPD70F4013M1GCA-UEU-G	(A)	POC
		μPD70F4013M1GCA9-UEU-G	(A9)	
		μPD70F4013M2GCA-UEU-G	(A)	No POC
		μPD70F4013M2GCA9-UEU-G	(A9)	
SG4-H-1.5M	μ PD70F4014	μPD70F4014M1GCA-UEU-G	(A)	POC
		μPD70F4014M1GCA9-UEU-G	(A9)	
		μPD70F4014M2GCA-UEU-G	(A)	No POC
		μPD70F4014M2GCA9-UEU-G	(A9)	
SJ4-H-1M	μ PD70F4015	μPD70F4015M1GJA-GAE-G	(A)	POC
		μPD70F4015M1GJA9-GAE-G	(A9)	
		μPD70F4015M2GJA-GAE-G	(A)	No POC
		μPD70F4015M2GJA9-GAE-G	(A9)	
SJ4-H-1.5M	μ PD70F4016	μPD70F4016M1GJA-GAE-G	(A)	POC
		μPD70F4016M1GJA9-GAE-G	(A9)	
		μPD70F4016M2GJA-GAE-G	(A)	No POC
		μPD70F4016M2GJA9-GAE-G	(A9)	
SK4-H-1.5M	μ PD70F4017	μPD70F4017M1GMA-GAR-G	(A)	POC
		μPD70F4017M1GMA9-GAR-G	(A9)	
		μPD70F4017M2GMA-GAR-G	(A)	No POC
		μPD70F4017M2GMA9-GAR-G	(A9)	
SK4-H-2M	μ PD70F4018	μPD70F4018M1GMA-GAR-G	(A)	POC
		μPD70F4018M1GMA9-GAR-G	(A9)	
		μPD70F4018M2GMA-GAR-G	(A)	No POC
		μPD70F4018M2GMA9-GAR-G	(A9)	

1.5 Pin Connection Diagram

1.5.1 V850E2/SG4-H

TBD

1.5.2 V850E2/SJ4-H

TBD

1.5.3 V850E2/SK4-H

<R> • 176-pin plastic LQFP (fine-pitch) (24 × 24)

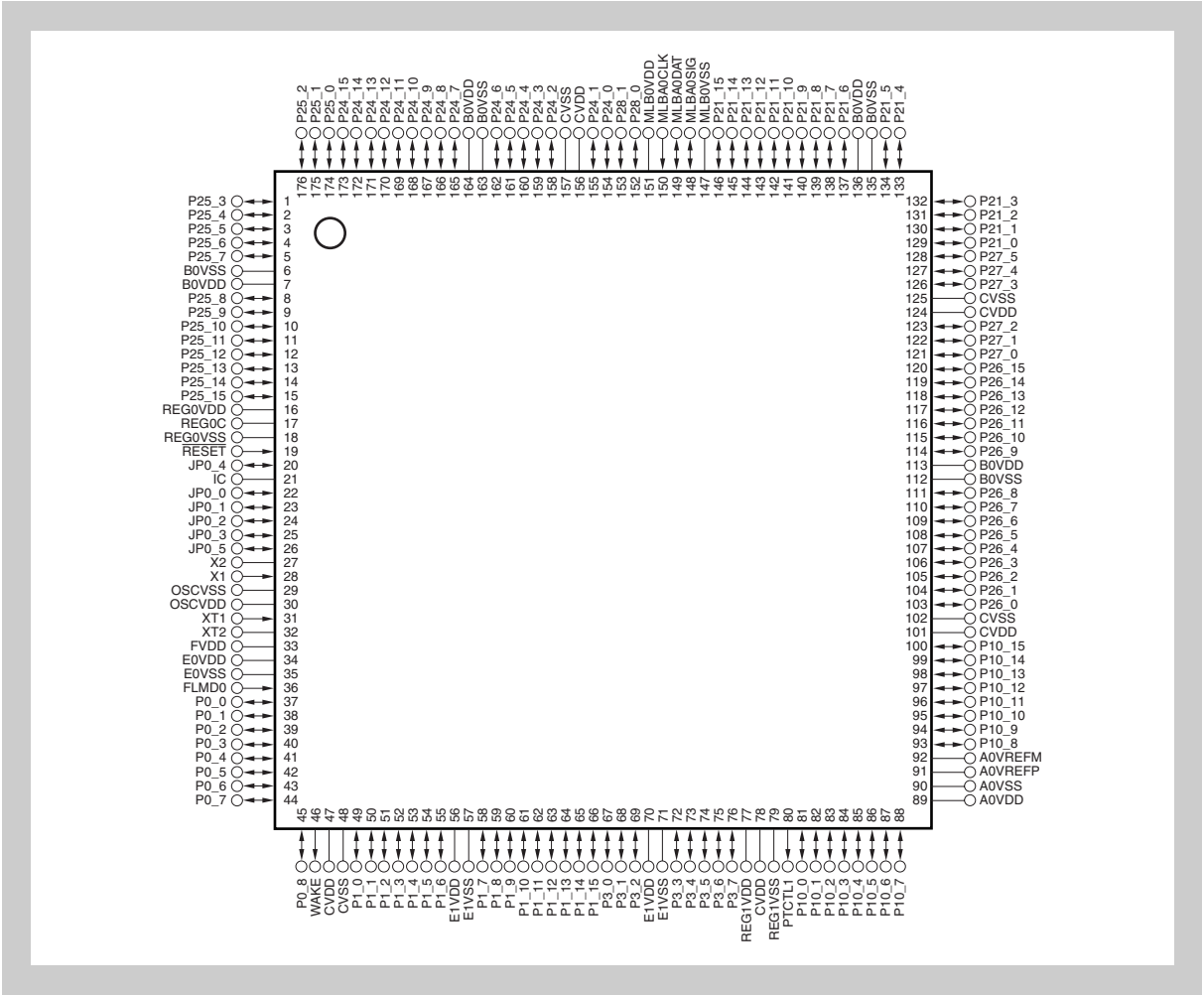


Table 1-6 Pin assignment (1/5)

Pin no.	Pin name
1	P25_3/MEMC0AD3/TAUA0I3/TAUA0O3/IISA0SDI/IISA0SDO/CSIH0RYI/CSIH0RYO
2	P25_4/MEMC0AD4/TAUA0I4/TAUA0O4/IISA1SDO/CSIH0SC
3	P25_5/MEMC0AD5/TAUA0I5/TAUA0O5/IISA1SCK
4	P25_6/MEMC0AD6/TAUA0I6/TAUA0O6/IISA1WS
5	P25_7/MEMC0AD7/TAUA0I7/TAUA0O7/IISA1SDI/IISA1SDO
6	B0VSS
7	B0VDD
8	P25_8/MEMC0AD8/TAUA0I8/TAUA0O8/IISA2SDO/$\overline{\text{CSIH1SSI}}$
9	P25_9/MEMC0AD9/TAUA0I9/TAUA0O9/IISA2SCK/CSIH1SI
10	P25_10/MEMC0AD10/TAUA0I10/TAUA0O10/IISA2WS/CSIH1SO
11	P25_11/MEMC0AD11/TAUA0I11/TAUA0O11/IISA2SDI/IISA2SDO/CSIH1RYI/CSIH1RYO
12	P25_12/MEMC0AD12/TAUA0I12/TAUA0O12/IISA3SDO/CSIH1SC
13	P25_13/MEMC0AD13/TAUA0I13/TAUA0O13/IISA3SCK
14	P25_14/MEMC0AD14/TAUA0I14/TAUA0O14/IISA3WS
15	P25_15/MEMC0AD15/TAUA0I15/TAUA0O15/IISA3SDI/IISA3SDO

Table 1-6 Pin assignment (2/5)

Pin no.	Pin name
16	REG0VDD
17	REG0C
18	REG0VSS
19	$\overline{\text{RESET}}$
20	JP0_4/$\overline{\text{DCUTRST}}$
<R> 21	IC[a]
22	JP0_0/INTP0/TAUJ0I0/TAUJ0O0/DCUTDI
23	JP0_1/INTP1/TAUJ0I1/TAUJ0O1/DCUTDO
24	JP0_2/INTP2/TAUJ0I2/TAUJ0O2/DCUTCK
25	JP0_3/INTP3/TAUJ0I3/TAUJ0O3/DCUTMS
26	JP0_5/NMI/RTCA0OUT/$\overline{\text{DCURDY}}$
27	X2
28	X1
29	OSCVSS
30	OSCVDD
31	XT1
32	XT2
<R> 33	FVDD
34	E0VDD
35	E0VSS
36	FLMD0
37	P0_0/TAUJ0I0/TAUJ0O0/INTP4/$\overline{\text{CSIG0SSI}}$/IICB0SDA/$\overline{\text{RESETOUT}}$
38	P0_1/TAUJ0I1/TAUJ0O1/INTP5/CSIG0SO/IICB0SCL/FLMD1
39	P0_2/TAUJ0I2/TAUJ0O2/INTP6/CSIG0SI/IICB1SDA
40	P0_3/TAUJ0I3/TAUJ0O3/INTP7/CSIG0SC/IICB1SCL
41	P0_4/$\overline{\text{IEBB0RX}}$/FCN0TX/INTP8/CSIH1SI/URTE0TX/IICB2SDA
42	P0_5/FCN0RX/$\overline{\text{IEBB0TX}}$/INTP9/URTE0RX/CSIH1SO/IICB2SCL
43	P0_6/FCN1TX/INTP10/CSIH1RYI/CSIH1RYO/$\overline{\text{CSIH1SSI}}$/URTE10TX
44	P0_7/FCN1RX/INTP11/CSIH1SC/URTE10RX
45	P0_8
<R> 46	WAKE
47	CVDD
48	CVSS
49	P1_0/TAUA0I0/TAUA0O0/IISAACK/URTE1RX/ETH0RXER/CSIH0CSS0
50	P1_1/TAUA0I1/TAUA0O1/IISA0SCK/$\overline{\text{CSIG4SSI}}$/URTE1TX/ETH0RXD0/CSIH0CSS1
51	P1_2/TAUA0I2/TAUA0O2/IISA0WS/CSIG4SO/ETH0RXD1/CSIH0CSS2
52	P1_3/TAUA0I3/TAUA0O3/IISA0SDI/IISA0SDO/CSIG4SI/ETH0RXD2/CSIH0CSS3
53	P1_4/TAUA0I4/TAUA0O4/INTP12/IISA1SDO/CSIG4SC/ETH0RXD3/CSIH0CSS4
54	P1_5/TAUA0I5/TAUA0O5/IISA1SCK/ENCA0AIN/ETH0TXD0/$\overline{\text{CSIH0SSI}}$
55	P1_6/TAUA0I6/TAUA0O6/IISA1WS/ENCA0BIN/ETH0TXD1/CSIH0SI
56	E1VDD

Table 1-6 Pin assignment (3/5)

Pin no.	Pin name
57	E1VSS
58	P1_7/TAUA0I7/TAUA0O7/IISA1SDI/IISA1SDO/ENCA0ZIN/ETH0TXD2/CSIH0SO
59	P1_8/TAUA0I8/TAUA0O8/INTP13/IISA2SDO/ENCA0TIN0/ETH0TXD3/CSIH0RYI/CSIH0RYO
60	P1_9/TAUA0I9/TAUA0O9/IISA2SCK/ENCA0TIN1/ETH0TXEN/CSIH0SC
61	P1_10/TAUA0I10/TAUA0O10/IISA2WS/ENCA1AIN/CSIH0CSS5/URTE2RX/ETH0MDC
62	P1_11/TAUA0I11/TAUA0O11/IISA2SDI/IISA2SDO/ENCA1BIN/CSIH0CSS6/ETH0CRSDV/URTE2TX
63	P1_12/TAUA0I12/TAUA0O12/INTP14/IISA3SDO/ENCA1ZIN/CSIH0CSS7/ETH0MDI/ETH0MDO
64	P1_13/TAUA0I13/TAUA0O13/PCM0CLK/ENCA1TIN0/IICB1SCL
65	P1_14/TAUA0I14/TAUA0O14/PCM0SEN/ENCA1TIN1/URTE3TX/IICB3SDA
66	P1_15/TAUA0I15/TAUA0O15/PCM0SI/PCM0SO/URTE3RX/IICB3SCL
67	P3_0/TAUB2I1/TAUB2O1/PCM1CLK/KR0I0/CSIH1CSS0/IICB0SDA
68	P3_1/TAUB2I2/TAUB2O2/PCM1SEN/KR0I1/CSIH1CSS1/IICB0SCL
69	P3_2/TAUB2I3/TAUB2O3/PCM1SI/PCM1SO/KR0I2/CSIH1CSS2/IICB1SDA
70	E1VDD
71	E1VSS
72	P3_3/TAUB2I5/TAUB2O5/IISAACK/KR0I3/CSIH1CSS3/ETH0REFCLK/ETH0TXER
73	P3_4/TAUB2I6/TAUB2O6/INTP15/IISA3SDO/KR0I4/CSIH1CSS4/ETH0COL
74	P3_5/TAUB2I7/TAUB2O7/IISA3SCK/KR0I5/CSIH1CSS5/ETH0TXCLK
75	P3_6/TAUB2I9/TAUB2O9/IISA3WS/KR0I6/CSIH1CSS6/ETH0RXDV
76	P3_7/TAUB2I10/TAUB2O10/IISA3SDI/IISA3SDO/KR0I7/CSIH1CSS7/ETH0RXCLK
77	REG1VDD
78	CVDD
79	REG1VSS
80	PTCTL1
81	P10_0/ADCA0I0
82	P10_1/ADCA0I1
83	P10_2/ADCA0I2
84	P10_3/ADCA0I3
85	P10_4/ADCA0I4
86	P10_5/ADCA0I5
87	P10_6/ADCA0I6
88	P10_7/ADCA0I7
89	A0VDD
90	A0VSS
<R> 91	A0VREFP
<R> 92	A0VREFM
93	P10_8/ADCA0I8
94	P10_9/ADCA0TRG0/ADCA0I9
95	P10_10/ADCA0TRG1/ADCA0I10
96	P10_11/ADCA0TRG2/ADCA0I11
97	P10_12/ADCA0I12

Table 1-6 Pin assignment (4/5)

Pin no.	Pin name
98	P10_13/ADCA0I13
99	P10_14/ADCA0I14
100	P10_15/ADCA0I15
101	CVDD
102	CVSS
103	P26_0/KR0I0/MEMC0A0/TAUB2I0/TAUB2O0/IISA4SDO/$\overline{\text{CSIH2SSI}}$
104	P26_1/KR0I1/MEMC0A1/TAUB2I1/TAUB2O1/IISA4SCK/CSIH2SI
105	P26_2/KR0I2/MEMC0A2/TAUB2I2/TAUB2O2/IISA4WS/CSIH2SO
106	P26_3/KR0I3/MEMC0A3/TAUB2I3/TAUB2O3/IISA4SDI/IISA4SDO/CSIH2RYI/CSIH2RYO
107	P26_4/KR0I4/MEMC0A4/TAUB2I4/TAUB2O4/IISA5SDO/CSIH2SC
108	P26_5/KR0I5/MEMC0A5/TAUB2I5/TAUB2O5/IISA5SCK/CSIH2CSS0
109	P26_6/KR0I6/MEMC0A6/TAUB2I6/TAUB2O6/IISA5WS/CSIH2CSS1
110	P26_7/KR0I7/MEMC0A7/TAUB2I7/TAUB2O7/IISA5SDI/IISA5SDO/CSIH2CSS2
111	P26_8/INTP8/MEMC0A8/TAUB2I8/TAUB2O8/IISA3SCK/CSIH2CSS3
112	B0VSS
113	B0VDD
114	P26_9/INTP9/MEMC0A9/TAUB2I9/TAUB2O9/IISA3WS/CSIH2CSS4
115	P26_10/INTP10/MEMC0A10/TAUB2I10/TAUB2O10/IISA2SCK/CSIH2CSS5
116	P26_11/INTP11/MEMC0A11/TAUB2I11/TAUB2O11/IISA2WS/CSIH2CSS6
117	P26_12/INTP12/MEMC0A12/TAUB2I12/TAUB2O12/IISA0SCK/CSIH2CSS7
118	P26_13/INTP13/MEMC0A13/TAUB2I13/TAUB2O13/IISA0WS
119	P26_14/INTP14/MEMC0A14/TAUB2I14/TAUB2O14/IISA5SDI/IISA5SDO
120	P26_15/INTP15/MEMC0A15/TAUB2I15/TAUB2O15/IISA4SDI/IISA4SDO
121	P27_0/INTP0/MEMC0A16/IISA3SDI/IISA3SDO/CSIH1CSS0
122	P27_1/INTP1/MEMC0A17/IISA2SDI/IISA2SDO/CSIH1CSS1
123	P27_2/INTP2/MEMC0A18/IISA1SDI/IISA1SDO/CSIH1CSS2
124	CVDD
125	CVSS
126	P27_3/INTP3/MEMC0A19/IISA0SDI/IISA0SDO/CSIH1CSS3
127	P27_4/INTP4/MEMC0A20/IISA0SCK/URTE10TX
128	P27_5/INTP5/MEMC0A21/IISA0WS/URTE10RX
129	P21_0/$\overline{\text{MEMC0BEN3}}$/MEMC0DQM3/ETH0COL
130	P21_1/$\overline{\text{MEMC0BEN2}}$/MEMC0DQM2/ETH0CRSDV
131	P21_2/$\overline{\text{MEMC0BEN1}}$/MEMC0DQM1
132	P21_3/$\overline{\text{MEMC0BEN0}}$/MEMC0DQM0
133	P21_4/$\overline{\text{MEMC0WR}}$/$\overline{\text{IEBB0RX}}$/URTE2TX/$\overline{\text{CSIG0SSI}}$/$\overline{\text{MEMC0WE}}$/IICB0SDA
134	P21_5/$\overline{\text{MEMC0RD}}$/URTE2RX/$\overline{\text{IEBB0TX}}$/CSIG0SO/IICB0SCL
135	B0VSS
136	B0VDD
137	P21_6/MEMC0CLK/URTE1TX/CSIG0SI/IICB1SDA
138	P21_7/$\overline{\text{MEMC0WAIT}}$/URTE1RX/CSIG0SC/IICB1SCL

Table 1-6 Pin assignment (5/5)

Pin no.	Pin name
139	P21_8/$\overline{\text{MEMC0SDRAS}}$/IICB3SDA
140	P21_9/$\overline{\text{MEMC0CS2}}$/$\overline{\text{MEMC0SDCAS}}$/IICB3SCL
141	P21_10/$\overline{\text{MEMC0CS3}}$/FCN1RX
142	P21_11/$\overline{\text{MEMC0CS4}}$/FCN1TX
143	P21_12/$\overline{\text{MEMC0CS7}}$/$\overline{\text{CSIG4SSI}}$
144	P21_13/$\overline{\text{MEMC0HLDRQ}}$/$\overline{\text{MEMC0DSTB}}$/FCN0RX/MEMC0CKE/CSIG4SO/IICB2SDA
145	P21_14/$\overline{\text{MEMC0HLDAK}}$/URTE0RX/FCN0TX/CSIG4SI/IICB2SCL
146	P21_15/$\overline{\text{MEMC0ASTB}}$/URTE0TX/CSIG4SC
147	MLB0VSS
148	MLBA0SIG
149	MLBA0DAT
150	MLBA0CLK
151	MLB0VDD
152	P28_0/INTP6/MEMC0A22/MEMC0A24/IISA1SCK
153	P28_1/INTP7/MEMC0A23/MEMC0A25/IISA1WS
154	P24_0/MEMC0AD16/INTP0/CSIH0CSS0/ENCA0AIN/ETH0RXER
155	P24_1/MEMC0AD17/INTP1/CSIH0CSS1/ENCA0BIN/ETH0RXD0
156	CVDD
157	CVSS
158	P24_2/MEMC0AD18/INTP2/CSIH0CSS2/ENCA0ZIN/ETH0RXD1
159	P24_3/MEMC0AD19/INTP3/CSIH0CSS3/ENCA0TIN0/ETH0RXD2
160	P24_4/MEMC0AD20/INTP4/CSIH0CSS4/ENCA0TIN1/ETH0RXD3
161	P24_5/MEMC0AD21/INTP5/CSIH0CSS5/ETH0TXD0
162	P24_6/MEMC0AD22/INTP6/CSIH0CSS6/ETH0TXD1
163	B0VSS
164	B0VDD
165	P24_7/MEMC0AD23/INTP7/CSIH0CSS7/ETH0TXD2
166	P24_8/MEMC0AD24/INTP8/CSIH1CSS0/ENCA1AIN/ETH0TXD3
167	P24_9/MEMC0AD25/INTP9/CSIH1CSS1/ENCA1BIN/ETH0TXEN
168	P24_10/MEMC0AD26/INTP10/CSIH1CSS2/ENCA1ZIN/ETH0MDC
169	P24_11/MEMC0AD27/INTP11/CSIH1CSS3/ENCA1TIN0/ETH0REFCLK/ETH0TXER
170	P24_12/MEMC0D28/INTP12/CSIH1CSS4/ENCA1TIN1/ETH0MDI/ETH0MDO
171	P24_13/MEMC0D29/INTP13/CSIH1CSS5/ETH0TXCLK
172	P24_14/MEMC0D30/INTP14/CSIH1CSS6/ETH0RXDV
173	P24_15/MEMC0D31/INTP15/CSIH1CSS7/ETH0RXCLK
174	P25_0/MEMC0AD0/TAUA0I0/TAUA0O0/IISAACK/IISA0SDO/$\overline{\text{CSIH0SSI}}$
175	P25_1/MEMC0AD1/TAUA0I1/TAUA0O1/IISA0SCK/CSIH0SI
176	P25_2/MEMC0AD2/TAUA0I2/TAUA0O2/IISA0WS/CSIH0SO

<R> [a)] Be sure to input a low-level voltage to the IC pin.

Chapter 2 Pin Functions

This chapter provides a general description of the pin functions.

The first section describes all specifications specific to the pins, such as pin groups and register base addresses.

The second section describes the features provided by each port.

The third section provides a summary of the features of each pin in the V850E2/Sx4-H.

2.1 Features

Port group

The V850E2/Sx4-H provides the following port groups, indicated by the numbers in the table below.

Table 2-1 Port groups in V850E2/Sx4-H

Port group	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of groups	7	10	11
Name	P0, P1, P10, P21, P25, P27, JP0	P0, P1, P3, P10, P21, P25 to P28, JP0	P0, P1, P3, P10, P21, P24 to P28, JP0

Port group index n

Throughout this chapter, the port groups are identified by using the index "n" (n = 0, 1, 3, 10, 21, 24 to 28, JP0). For example, the port mode control register of the Pn pin is PMCn.

Register addresses

The addresses of the registers used to control the general ports and JTAG ports are given as addresses offset from the base addresses <PORTn_base>and <JPORTn_base>.

The base addresses <PORTn_base> and <JPORTn_base> are listed in the following table:

Table 2-2 Port base address <PORTn_base> and <JPORTn_base>

<PORTn_base> address	<JPORTn_base> address
FF40 0000_H	FF44 0000_H

2.2 Overview

The microcontroller has various pins for input/output functions, known as ports. The ports are organized in port groups.

The V850E2/Sx4-H also has several control registers to enable pins to be used as other than general purpose input/output pins.

For a description of the terms pin, port, or port group, see *2.2.1 "Terms" on page 45*.

Functional overview

- Pins can be set individually
- The following features can be selected for most of the pins:
  - 4 types of input buffer characteristics
  - 2 types of output buffer characteristics
  - Open-drain emulation
  - Pull-up or pull-down resistor connection
- The following registers are provided for most of the ports:
  - Register for reading the pin values
  - Port register
  - Port set/reset register
  - Register for inverting the output

2.2.1 Terms

The following terms are used in this section:

- **Pin**

  Denotes the physical pin. Every pin is denoted by a unique pin number.
  A pin can be used in several modes. Each pin is assigned a name that reflects its function, which is determined by the selected mode.

- **Port group**

  Denotes a group of pins. All the pins of a specific port group are controlled by the same port control register.

- **Port mode and ports**

  A pin in port mode works as a general purpose input/output pin. It is then called “port”.
  The corresponding name is Pn_m. For example, P0_7 denotes port 7 of port group 0. It is referenced as “port P0_7”.

- **Alternative mode**

  In alternative mode, a pin can be used for various non-general-purpose input/output functions, such as the input/output pin of on-chip peripherals.
  The corresponding pin name depends on the selected function. For example, pin INTP0 denotes the pin for one of the external interrupt inputs.
  Note that two different names can refer to the same physical pin, for example P0_0 and INTP0. The different names indicate the function of the pin at that time.

- **Port type**

  The port controller is determined by a setting register specification. The type of each different controller is known as the port type.

JTAG ports The JTAG port groups are used for connecting a debugger for on-chip debugging. These are special port groups provided because the microcontroller cannot be used for the user's application while on-chip debugging is being executed. When a debugger is not connected and the microcontroller is operating normally, these port groups can be used in the same way as the other port groups.

JTAG port group registers and bit names are prefixed by a "J". For example, JP0 denotes JTAG port group 0, and JPMn.JPMnm denotes the JPMnm bit of the JPMn port mode control register.

Note In this chapter, all descriptions of ports and registers also apply to JTAG ports and registers, unless otherwise specified.

2.2.2 Overview of pin functions

Pins can operate in three modes.

- Port mode (PMCn.PMCnm bit = 0)

  A pin in port mode operates as a general purpose input/output pin. The I/O mode is selected by setting the PMn.PMnm bit.

- Software I/O control alternative mode (PMCn.PMCnm bit = 1, PIPCn.PIPCnm bit = 0)

  In this mode, the pins operate as alternative functions. The I/O mode is selected by setting the PMn.PMnm bit by using software.

- Direct I/O control alternative mode (PMCn.PMCnm bit = 1, PIPCn.PIPCnm bit = 1)

  In this mode, the pins operate as alternative functions. Unlike the software I/O alternative mode, however, the I/O mode is selected by the alternative function.

An overview of the register settings is given in the tables below.

Table 2-3 Pin function configuration (overview)

Mode	Bit			I/O
	PMCnm	PMnm	PIPCnm	
Port mode	0	0	X	O
		1[a]		I
Software I/O control alternative mode	1	0	0	O
		1	0	I
Direct I/O control alternative mode		X	1	Controlled by the alternative function

a) The input buffer must be enabled (PIBCnm = 1).

If a pin is in alternative mode (PMCn.PMCnm = 1), one of up to four alternative functions can be selected for that pin by using the PFCn and PFCEn registers.

- Software I/O control alternative mode (PIPCn.PIPCnm = 0)
  - Output (PMnm = 0): ALT-OUT1 to ALT-OUT4
  - Input (PMnm = 1): ALT-IN1 to ALT-IN4
- Direct I/O control alternative mode (PIPCn.PIPCnm = 1)
  - The I/O mode for ALT-OUT1 to ALT-OUT4 and ALT-IN1 to ALT-IN4 is directly selected by the alternative function.

Table 2-4 Alternative mode selection overview (PMCn.PMCnm bit = 1)

Mode	Register				I/O
	PIPC[a]	PM[a]	PFCE	PFC	
Alternative-function output mode 1 (ALT-OUT1)	0	0	0	0	O
Alternative-function input mode 1 (ALT-IN1)		1			I
Alternative-function output mode 2 (ALT-OUT2)		0	0	1	O
Alternative-function input mode 2 (ALT-IN2)		1			I
Alternative-function output mode 3 (ALT-OUT3)		0	1	0	O
Alternative-function input mode 3 (ALT-IN3)		1			I
Alternative-function output mode 4 (ALT-OUT4)		0	1	1	O
Alternative-function input mode 4 (ALT-IN4)		1			I

a) If PIPCn.PIPCnm = 1, the I/O direction is directly controlled by the peripheral (alternative function) and PM is ignored.

If a pin is in alternative mode (PMCn.PMCnm = 1), one of up to four alternative functions can be selected for that pin by using the PFCn and PFCEn registers.

2.2.3 Pin data input/output

The registers used for data input/output are described below.

The location that is read via the PPRn register differs depending on the pin mode.

Output data In the port mode (PMCn.PMCnm = 0), the value of the Pn.Pnm bit is output from the Pn_m pin.

Input data When the PPRn register is read, either the value of the Pn_m pin, the value of the corresponding bit of the port register Pn.Pnm, or the value output by the alternative function is returned.

Which value is returned depends on the pin mode and setting of several control bits.

The different PPRn read modes are shown in the table below.

Table 2-5 PPRnm read values

PMC nm	PM nm	PIBC nm	PIPC nm	PODC nm	Mode	PPRnm read value
0	1	0	X	X	Port input, input buffer disabled	Pn.Pnm register
		1		X	Port input, input buffer enabled	Pn_m pin
	0	X		0	Port push-pull output	Pn.Pnm register[a]
				1	Port open-drain output	
1	1	X	0	X	Software I/O control alternative-function input	Pn_m pin
	0			0	Software I/O control alternative-function push-pull output	Alternative-function internal output signal[a]
				1	Software I/O control alternative-function open-drain output	
	X		1	0	Direct I/O control alternative-function push-pull output	I/O port in alternative mode: • Input: Pn_m pin • Output: Alternative-function internal output signal[a]
				1	Direct I/O control alternative-function open-drain output	

a) When PBDCnm = 1, the level of the Pn_m pin is returned by the PPRnm register.

The control registers in the above table have the following effects:

- PMCn.PMCnm bit

  This bit selects port mode (PMCnm = 0) or alternative mode (PMCnm = 1).

- PMn.PMnm bit

  This bit selects input or output when the port mode (PMCnm = 0) and software I/O control alternative mode (PMCnm = 1, PIPCnm = 0) have been selected.

- PIBCn.PIBCnm bit

  This bit disables (PIBCnm = 0) or enables (PIBCnm = 1) the input buffer in input port mode (PMCnm = 0 and PMnm = 1). If the input buffer is disabled, PPRnm reads the Pn.Pnm bit, otherwise the Pn_m pin level is returned.

- PIPCn.PIPCnm bit

  This bit selects software I/O control alternative-function mode or direct I/O control alternative-function mode.

- PODCn.PODnm bit

  This bit selects push-pull output (PODCnm = 0) or open-drain output (PODCnm = 1).

- PBDCn.PBDCnm bit

  When this bit is set to 1, PPRnm is commanded to read the level of the Pn_m pin. In other words, if the port is in output mode, the bidirectional mode, in which the level of the Pn_m pin can be read, is enabled.

Caution When using Pn_m as an alternative function (PMCn.PMCnm = 1, PMn.PMnm = 0), the level of the Pn_m pin can be read at the PPRn.PPRnm bit by enabling bidirectional mode PBDCn.PBDCnm = 1).

Note, however, that in this case, the level of the Pn_m pin will be input to the alternative function that the Pn_m pin is being used as.

Writing to the Pn register

The data to be output via port Pn_m in port mode (PMCn.PMCnm = 0) is held in port register Pn.

Pn data can be overwritten in two ways:

- By writing data directly to the Pn register.

  In this case, new data can be written directly to the Pn register.

- By performing an indirect bitwise operation (a "set", "reset", or "not" operation) on the Pn register.

  An indirect bitwise operation ("set", "reset", or "not"), can be performed on the Pn register by using the following two registers:

  – Port set reset register PSRn

    If PSRn.PSRn(m + 16) = 1, the value of the Pn.Pnm bit is determined by the value of the PSRn.PSRnm bit.

    In other words, the Pnm bit can be set or reset without writing directly to the Pn register.

  – Port NOT register PNOTn

    By setting PNOTn.PNOTnm to 1, the Pn.Pnm bit can be inverted without writing directly to the Pn register.

  An indirect bitwise operation on the Pn register ("set", "reset", or "not"), has no effect on the bits that do not need to be updated, allowing you to overwrite only the bit or bits that need to be overwritten.

2.2.4 Port control logic diagram

The diagram below shows the port control logic.

Caution The logic shown in this diagram is for reference only. This diagram does not represent the actual circuits.

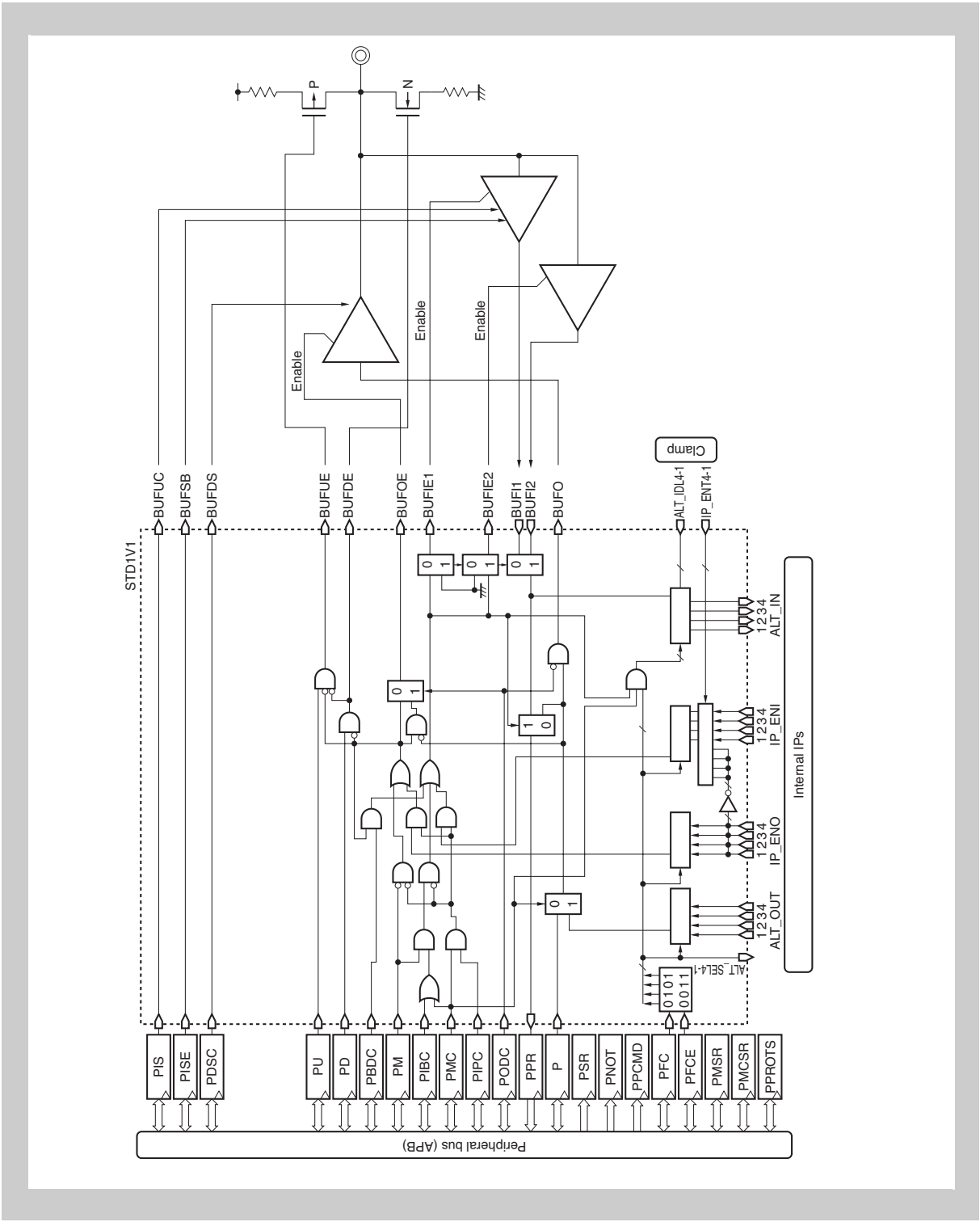


Figure 2-1 Port control logic diagram

2.2.5 Writing to protected registers

Write-protected registers are protected from being accessed by mistake such as by erroneous program execution.

The following port registers have special write protection features:

- Port drive strength control registers PDSCn and JPDSCn
- Port open drain control registers PODCn and JPODCn

(1) Port register protection clusters

The protected registers in port group n are grouped together as a port register protection cluster.

The port groups whose registers are grouped in port register protection clusters are described in the section Port register protection clusters.

(2) Port protection unlock sequence

Write-protected registers can only be written by using a special protection unlock sequence.

1. Write the fixed value $A5_H$ to the protection command register PPCMDn.
2. Write the desired value to the protected register.
3. Write the bitwise inversion of the desired value to the protected register.
4. Write the desired value to the protected register.
5. Verify successful writing of the desired value to the protected register by verifying that PPROTSn.PPROTSn_0 = 0. If the PPROTSn.PPROTSn_0 bit is 1, start again from step 1.

If another register is accessed between steps 1 and 4, the protection mechanism behaves as follows:

- If the second register belongs to the same cluster, the write to the protected register fails (the PPROTSn.PPROTSn_0 bit indicates 1). In this case, start again from step 1.
- If the second register does not belong to the same cluster, the protection unlock sequence is not disrupted and the write to the first register can be completed successfully.

If the protection unlock sequence is interrupted, the protection mechanism behaves as follows:

- Interrupt during protection unlock sequence

  If the program branches to interrupt processing during the above protection unlock sequence and if the interrupt servicing routine does not access any protected register in the same port register protection cluster as the cluster whose protection is currently being unlocked, the protection unlock sequence is not disrupted and the write to the protected register can be completed successfully after the program returns from interrupt processing.

- Break by emulator during protection unlock sequence

  If the emulator generates a break during the above protection unlock sequence due, for example, to a breakpoint hit, register protection unlocking is suspended until after the program returns from the break and normal operation resumes.

  This means that even if a clock or standby control register is accessed during a break, the protection unlock sequence is undisturbed.

  That is, even if a clock or standby control register is accessed, the PROTSm.PROTERR bit is not set to 1.

2.3 Port Group Configuration Registers

This section starts with an overview of all configuration registers and then describes all registers in detail. The configuration registers are grouped as follows:

- *2.3.2 “Pin function configuration” on page 55*
- *2.3.3 “Pin data input/output” on page 62*
- *2.3.4 “Configuration of electrical characteristics” on page 66*

2.3.1 Overview

The following registers are used for setting the individual pins of the port groups:

Table 2-6 Port group configuration registers (1/2)

Register name	Symbol	Address
Pin function setting		
Port mode control register	PMCn	<PORTn_base> + 0400_H + n x 4
	JPMCn	<JPORTn_base> + 0040_H + n x 4
Port mode control set/reset register	PMCSRn	<PORTn_base> + 0900_H + n x 4
	JPMCSRn	<JPORTn_base> + 0090_H + n x 4
Port IP control register	PIPCn	<PORTn_base> + 4200_H + n x 4
Port mode register	PMn	<PORTn_base> + 0300_H + n x 4
	JPMn	<JPORTn_base> + 0030_H + n x 4
Port mode set/reset register	PMSRn	<PORTn_base> + 0800_H + n x 4
	JPMSRn	<JPORTn_base> + 0080_H + n x 4
Port input buffer control register	PIBCn	<PORTn_base> + 4000_H + n x 4
	JPIBCn	<JPORTn_base> + 0400_H + n x 4
Port function control register	PFCn	<PORTn_base> + 0500_H + n x 4
	JPFCn	<JPORTn_base> + 0050_H + n x 4
Port function control expansion register	PFCEn	<PORTn_base> + 0600_H + n x 4
Pin data input/output		
Port bi-direction control register	PBDCn	<PORTn_base> + 4100_H + n x 4
	JPBDCn	<JPORTn_base> + 0410_H + n x 4
Port pin read register	PPRn	<PORTn_base> + 0200_H + n x 4
	JPPRn	<JPORTn_base> + 0020_H + n x 4
Port register	Pn	<PORTn_base> + 0000_H + n x 4
	JPn	<JPORTn_base> + 0000_H + n x 4
Port NOT register	PNOTn	<PORTn_base> + 0700_H + n x 4
	JPNOTn	<JPORTn_base> + 0070_H + n x 4
Port set/reset register	PSRn	<PORTn_base> + 0100_H + n x 4
	JPSRn	<JPORTn_base> + 0010_H + n x 4

Table 2-6 Port group configuration registers (2/2)

Register name	Symbol	Address
Specifying electrical characteristics		
Pull-up option register	PUn	<PORTn_base> + 4300_H + n x 4
	JPUn	<JPORTn_base> + 0430_H + n x 4
Pull-down option register	PDn	<PORTn_base> + 4400_H + n x 4
	JPDn	<JPORTn_base> + 0440_H + n x 4
Port drive strength control register	PDSCn	<PORTn_base> + 4600_H + n x 4
	JPDSCn	<JPORTn_base> + 0460_H + n x 4
Port open drain control register	PODCn	<PORTn_base> + 4500_H + n x 4
	JPODCn	<JPORTn_base> + 0450_H + n x 4
Port input buffer selection register	PISn	<PORTn_base> + 4700_H + n x 4
	JPISn	<JPORTn_base> + 0470_H + n x 4
Port input buffer selection expansion register	PISEn	<PORTn_base> + 4800_H + n x 4
	JPISEn	<JPORTn_base> + 0480_H + n x 4
Port register protection		
Port register protection command register	PPCMDn	<PORTn_base> + $4C00_H$ + n x 4
	JPPCMDn	<JPORTn_base> + $04C0_H$ + n x 4
Port protection status register	PPROTSn	<PORTn_base> + $4B00_H$ + n x 4
	JPPROTSn	<JPORTn_base> + $04B0_H$ + n x 4

<PORTn_base> The base addresses of PORTn < PORTn_base> is defined in *2.1 "Features"* under the key word *"Register addresses"*.

JTAG port registers JTAG port registers are not explicitly described in the following register descriptions.

All descriptions (except for those of the PFCE register) apply to JTAG port registers. Note, however, that the JTAG port register base address differs from that of regular ports.

<JPORTn_base> The base addresses of JPORTn < JPORTn_base> is defined in *2.1 "Features"* under the key word *"Register addresses"*.

Register default value The default values of the registers after reset depend on the ports. The default values are not described here, rather they are described in *2.4 "V850E2/Sx4-H Port Group Configuration"*.

2.3.2 Pin function configuration

(1) PMCn - Port mode control register

This register is used to specify whether the individual pins of port group n are in port mode or in alternative mode.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 0400_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMCn15	PMCn15	PMCn13	PMCn12	PMCn11	PMCn10	PMCn9	PMCn8	PMCn7	PMCn6	PMCn5	PMCn4	PMCn3	PMCn2	PMCn1	PMCn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-7 PMCn register contents

Bit position	Bit name	Function
15 to 0	PMC[15:0]	Specifies the operation mode of the corresponding pin: 0: Port mode 1: Alternative mode

(2) PMCSRn - Port mode control set and reset register

This register provides an alternative method for writing data to the PMCn register.

The higher bits of the PMCSRn register specify whether data can be written to the PMCn.PMCnm bit specified by the lower bits of the PMCSRn register.

Access This register can be read or written in 32-bit units.

Bits 31 to 16 are always read as 0000_H. Bits 15 to 0 are read as the value of the PMCn register.

Address <PORT_base> + 0900_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PMCSRn31	PMCSRn30	PMCSRn29	PMCSRn28	PMCSRn27	PMCSRn26	PMCSRn25	PMCSRn24	PMCSRn23	PMCSRn22	PMCSRn21	PMCSRn20	PMCSRn19	PMCSRn18	PMCSRn17	PMCSRn16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMCSRn15	PMCSRn14	PMCSRn13	PMCSRn12	PMCSRn11	PMCSRn10	PMCSRn9	PMCSRn8	PMCSRn7	PMCSRn6	PMCSRn5	PMCSRn4	PMCSRn3	PMCSRn2	PMCSRn1	PMCSRn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-8 PMCSRn register contents

Bit position	Bit name	Function
31 to 16	PMCSRn[31:16]	These are enable bits that specify whether the value of the corresponding lower bit of the PMCSRn register is written to the corresponding PPMCnm bit. 0: PMCnm does not depend on the PMCSRnm bit. 1: PMCnm becomes the value of the PMCSRnm bit. Example: If PMCSRn.PMCSRn31 = 1, the value of PMCSRn.PMCSRn15 is written to PMCn.PMCn15.
15 to 0	PMCSRn[15:0]	These are data bits that specify the value of the PMCnm bit if the corresponding higher bit, PMCSRn(m + 16), is 1. 0: PMCnm = 0 1: PMCnm = 1

(3) PIPCn - Port IP control register

This register is used to specify whether the I/O direction of the Pn_m pin is controlled by the PMnm bit of the port mode register (PMn) or by the alternative function.

If the Pn_m pin is in alternative mode (PMCn.PMCnm = 1), the PIPCn.PIPCnm bit must be set to 1 to enable the I/O direction of the Pn_m pin to be controlled by the alternative function.

By specifying this setting, the setting of the PMn.PMnm bit becomes invalid.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 4200_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIPCn15	PIPCn14	PIPCn13	PIPCn12	PIPCn11	PIPCn10	PIPCn9	PIPCn8	PIPCn7	PIPCn6	PIPCn5	PIPCn4	PIPCn3	PIPCn2	PIPCn1	PIPCn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-9 PIPCn register contents

Bit position	Bit name	Function
15 to 0	PIPC[15:0]	These bits specify the I/O mode. 0: The I/O mode is selected according to the value of the PMn.PMnm bit (software I/O control). 1: The I/O mode is selected by the alternative function (direct I/O control).

(4) PMn - Port mode register

The PMn register specifies whether the individual pins of port group n are in input mode or in output mode.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 0300_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMn15	PMn14	PMn13	PMn12	PMn11	PMn10	PMn9	PMn8	PMn7	PMn6	PMn5	PMn4	PMn3	PMn2	PMn1	PMn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-10 PMn register contents

Bit position	Bit name	Function
15 to 0	PMn[15:0]	Specifies input/output mode of the corresponding pin: 0: Output mode (output enabled) 1: Input mode (output disabled)

Notes

1. To use a port in input port mode (PMCn.PMCnm = 0 and PMn.PMnm = 1), the input buffer must be enabled (PIBCn.PIBCnm = 1).
2. By default, PMnm specifies the I/O direction in port mode (PMCn.PMCnm = 0) and alternative mode (PMCn.PMCnm = 1), because PIPCn.PIPCnm = 0 after reset.

(5) PMSRn - Port mode set and reset register

This register provides an alternative method for writing data to the PMn register.

The higher 16 bits of the PMSRn register specify whether data can be written to the PMn.PMnm bit specified by the lower 16 bits of the PMSRn register.

Access This register can be read or written in 32-bit units.

Bits 31 to 16 are always read as 0000_H. Bits 15 to 0 are read as the value of the PMn register.

Address <PORT_base> + 0800_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PMSRn31	PMSRn30	PMSRn29	PMSRn28	PMSRn27	PMSRn26	PMSRn25	PMSRn24	PMSRn23	PMSRn22	PMSRn21	PMSRn20	PMSRn19	PMSRn18	PMSRn17	PMSRn16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMSRn15	PMSRn14	PMSRn13	PMSRn12	PMSRn11	PMSRn10	PMSRn9	PMSRn8	PMSRn7	PMSRn6	PMSRn5	PMSRn4	PMSRn3	PMSRn2	PMSRn1	PMSRn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-11 PMSRn register contents

Bit position	Bit name	Function
31 to 16	PMSRn[31:16]	These are enable bits that specify whether the value of the corresponding lower bit of the PMSRn register is written to the corresponding PMCnm bit. 0: PMnm does not depend on the PMSRnm bit. 1: PMnm becomes the value of the PMSRnm bit. Example: If PMSRn.PMSRn31 = 1, the value of PMSRn.PMSRn15 is written to PMn.PMn15.
15 to 0	PMSRn[15:0]	These are data bits that specify the value of the PMnm bit if the corresponding higher bit, PMSRn(m+16), is 1. 0: PMnm = 0 1: PMnm = 1

(6) PIBCn - Port input buffer control register

In input port mode (PMCn.PMCnm = 0 and PMn.PMnm = 1) this register enables the port pin's input buffer.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 4000_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIBCn15	PIBCn14	PIBCn13	PIBCn12	PIBCn11	PIBCn10	PIBCn9	PIBCn8	PIBCn7	PIBCn6	PIBCn5	PIBCn4	PIBCn3	PIBCn2	PIBCn1	PIBCn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-12 PIBCn register contents

Bit position	Bit name	Function
15 to 0	PIBCn[15:0]	These bits enable or disable the input buffer. 0: Input buffer disabled 1: Input buffer enabled

Note When the input buffer is disabled, through current does not flow even when the pin level is Hi-Z. Thus the pin does not need to be fixed to a high or low level externally.

Caution Settings in this register are overruled in bi-directional mode (PBDCn.PBDCnm = 1).

(7) PFCn - Port function control register

This register, together with the PFCEn register, specifies the alternative function of the pins.

The I/O direction of several alternative functions can be controlled directly by using the Pn_m pin. For these alternative functions, the PIPCn.PIPCnm bit must be set to 1.

For all other alternative functions, the I/O direction is specified by using the PMn.PMnm bit.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 0500_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PFCn15	PFCn14	PFCn13	PFCn12	PFCn11	PFCn10	PFCn9	PFCn8	PFCn7	PFCn6	PFCn5	PFCn4	PFCn3	PFCn2	PFCn1	PFCn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-13 PFCn register contents

Bit position	Bit name	Function
15 to 0	PFCn[15:0]	Specifies the alternative function of a pin. For details, see *Table 2-4 "Alternative mode selection overview (PMCn.PMCnm bit = 1)"*.

(8) PFCEn - Port function control expansion register

This register, together with the PFCn register, specifies the alternative function of the pins.

The I/O direction of several alternative functions can be controlled directly by using the Pn_m pin. For these alternative functions, the PIPCn.PIPCnm bit must be set to 1.

For all other alternative functions, the I/O direction is specified by using the PMn.PMnm bit.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 0600_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PFCEn15	PFCEn14	PFCEn13	PFCEn12	PFCEn11	PFCEn10	PFCEn9	PFCEn8	PFCEn7	PFCEn6	PFCEn5	PFCEn4	PFCEn3	PFCEn2	PFCEn1	PFCEn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-14 PFCEn register contents

Bit position	Bit name	Function
15 to 0	PFCEn[15:0]	Specifies the alternative function of a pin. For details, see *Table 2-4 “Alternative mode selection overview (PMCn.PMCnm bit = 1)”*.

2.3.3 Pin data input/output

(1) PBDCn - Port bidirection control register

This register enables the input buffers, allowing the level of the Pn_m pin to always be read via the PPRn.PPRnm bit.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 4100_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PBDCn 15	PBDCn 14	PBDCn 13	PBDCn 12	PBDCn 11	PBDCn 10	PBDCn 9	PBDCn 8	PBDCn 7	PBDCn 6	PBDCn 5	PBDCn 4	PBDCn 3	PBDCn 2	PBDCn 1	PBDCn 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-15 PBDCn register contents

Bit position	Bit name	Function
15 to 0	PBDCn[15:0]	Enables/disables bidirectional mode of the corresponding pin: 0: Bidirectional mode disabled 1: Bidirectional mode enabled

Caution When using the Pn_m port as an alternative function (PMCn.PMCnm = 1, PMn.PMnm = 0), the level of the Pn_m pin can be read at the PPRn.PPRnm bit by enabling bidirectional mode (PBDCn.PBDCnm = 1). Note, however, that in this case, the level of the Pn_m pin will be input to the alternative function that the Pn_m pin is being used as.

Note If the PBDCnm bit is set to 1, the port mode setting of the PMn.PMnm bit becomes invalid.

(2) PPRn - Port pin read register

The PPRn register reflects the actual level of the Pn_m pin, the value of the Pn.Pnm bit, or the output level of the alternative function. The value read depends on various control settings as described in *Table 2-5 "PPRnm read values" on page 48*.

Access This register is read-only, in 16-bit units.

Address <PORT_base> + 0200_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPRn15	PPRn14	PPRn13	PPRn12	PPRn11	PPRn10	PPRn9	PPRn8	PPRn7	PPRn6	PPRn5	PPRn4	PPRn3	PPRn2	PPRn1	PPRn0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 2-16 PPRn register contents

Bit position	Bit name	Function
15 to 0	PPRn[15:0]	The value of the Pn_m pin, the value of the Pn.Pnm bit, or the output level of the alternative function

(3) Pn - Port register

In output port mode (PMCn.PMCnm = 0 and PMn.PMnm = 0), this register holds the Pn.Pnm data to be output via the related Pn_m port.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 0000_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pn15	Pn14	Pn13	Pn12	Pn11	Pn10	Pn9	Pn8	Pn7	Pn6	Pn5	Pn4	Pn3	Pn2	Pn1	Pn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-17 Pn register contents

Bit position	Bit name	Function
15 to 0	Pn[15:0]	These bits set the output level of the m pin (m = 0 to 15). 0: Outputs low level 1: Outputs high level

Note The bits of this register can be manipulated in several ways. See *"Writing to the Pn register"* in *2.2.3 "Pin data input/output" on page 7*.

(4) PNOTn - Port NOT register

This register allows a Pnm bit of the Pn port register to be inverted without directly writing to Pn.

Access This register can be read or written in 16-bit units. It is always read as 0000_H.

Address <PORT_base> + 0700_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PNOTn 15	PNOTn 14	PNOTn 13	PNOTn 12	PNOTn 11	PNOTn 10	PNOTn 9	PNOTn 8	PNOTn 7	PNOTn 6	PNOTn 5	PNOTn 4	PNOTn 3	PNOTn 2	PNOTn 1	PNOTn 0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 2-18 PNOTn register contents

Bit position	Bit name	Function
15 to 0	PNOTn[15:0]	Specifies whether to invert Pn.Pnm. 0: Pn.Pnm is not inverted (Pnm → Pnm) 1: Pn.Pnm is inverted ($\overline{Pnm}$ → Pnm)

(5) PSRn - Port set and reset register

This register provides an alternative method for writing data to the Pn register.

The higher 16 bits of the PSRn register specify whether data can be written to the Pn.Pnm bit specified by the lower 16 bits of the PSRn register.

Access This register can be read or written in 32-bit units.

Bits 31 to 16 are always read as 0000_H. Bits 15 to 0 are read as the value of the Pn register.

Address <PORT_base> + 0100_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PSRn31	PSRn30	PSRn29	PSRn28	PSRn27	PSRn26	PSRn25	PSRn24	PSRn23	PSRn22	PSRn21	PSRn20	PSRn19	PSRn18	PSRn17	PSRn16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSRn15	PSRn14	PSRn13	PSRn12	PSRn11	PSRn10	PSRn9	PSRn8	PSRn7	PSRn6	PSRn5	PSRn4	PSRn3	PSRn2	PSRn1	PSRn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-19 PSRn register contents

Bit position	Bit name	Function
31 to 16	PSRn[31:16]	These are enable bits that specify whether the value of the corresponding lower bit of the PSRnm register is written to the corresponding Pnm bit. 0: Pnm is independent of PSRnm 1: Pnm becomes the value of the PSRnm bit. Example: If PSRn.PSRn31 = 1, the value of PSRn.PSRn15 is written to Pn.Pn15.
15 to 0	PSRn[15:0]	These are data bits that specify the value of the Pnm bit if the corresponding higher bit, PSRn(m+16), is 1. 0: Pnm = 0 1: Pnm = 1

2.3.4 Configuration of electrical characteristics

(1) PUn - Pull-up option register

This register is used to specify whether a pull-up resistor is connected to an input pin.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 4300_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUn15	PUn14	PUn13	PUn12	PUn11	PUn10	PUn9	PUn8	PUn7	PUn6	PUn5	PUn4	PUn3	PUn2	PUn1	PUn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-20 PUn register contents

Bit position	Bit name	Function
15 to 0	PUn[15:0]	These bits specify whether a pull-up resistor is connected to the corresponding pin. 0: Do not connect pull-down resistor 1: Connect pull-down resistor

Notes
1. If a pin is configured so that both a pull-up resistor (PUn.PUnm = 1) and a pull-down resistor (PDn.PDnm = 1) are connected, the pull-down resistor is automatically selected and the pull-up resistor is not connected.
2. The pull-up resistor has no effect when the pin operates in output mode.

(2) PDn - Pull-down option register

This register is used to specify whether a pull-down resistor is connected to an input pin.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 4400_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDn15	PDn14	PDn13	PDn12	PDn11	PDn10	PDn9	PDn8	PDn7	PDn6	PDn5	PDn4	PDn3	PDn2	PDn1	PDn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-21 PDn register contents

Bit position	Bit name	Function
15 to 0	PDn[15:0]	These bits specify whether a pull-down resistor is connected to the corresponding pin. 0: Do not connect pull-down resistor 1: Connect pull-down resistor

Notes
1. If a pin is configured so that both a pull-up resistor (PUn.PUnm = 1) and a pull-down resistor (PDn.PDnm = 1) are connected, the pull-down resistor is automatically selected and the pull-up resistor is not connected.
2. The pull-down resistor has no effect when the pin operates in output mode.

(3) PDSCn - Port drive strength control register

This register is used to specify the output drive strength of each port pin. This feature also relates to the fast (high drive strength) and slow (low drive strength) modes of the output buffer.

This register is one of the OS registers of SPF.

Access This register can be read or written in 32-bit units.

This register must be updated by using the correct write sequence, which involves using the PPCMD register.

Address <PORT_base> + 4600_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PDSCn 31	PDSCn 30	PDSCn 29	PDSCn 28	PDSCn 27	PDSCn 26	PDSCn 25	PDSCn 24	PDSCn 23	PDSCn 22	PDSCn 21	PDSCn 20	PDSCn 19	PDSCn 18	PDSCn 17	PDSCn 16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDSCn 15	PDSCn 14	PDSCn 13	PDSCn 12	PDSCn 11	PDSCn 10	PDSCn 9	PDSCn 8	PDSCn 7	PDSCn 6	PDSCn 5	PDSCn 4	PDSCn 3	PDSCn 2	PDSCn 1	PDSCn 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

<R> **Table 2-22 PDSCn register contents**

Bit position	Bit name	Function
31 to 0	PDSCn[31:0]	These bits specify the port drive strength of the output buffer of each port pin. 0: Low drive strength (slow mode) 1: High drive strength (fast mode)

(4) PODCn - Port open drain control register

This register selects push-pull or open-drain as output buffer operation.

This register is one of the OS registers of SPF.

Access This register can be read or written in 32-bit units.

This register is write-protected, and can only be written by using a special instruction sequence. For details, see *2.3.5 "Port register protection" on page 70*.

Address <PORT_base> + 4500_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PODCn 31	PODCn 30	PODCn 29	PODCn 28	PODCn 27	PODCn 26	PODCn 25	PODCn 24	PODCn 23	PODCn 22	PODCn 21	PODCn 20	PODCn 19	PODCn 18	PODCn 17	PODCn 16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PODCn 15	PODCn 14	PODCn 13	PODCn 12	PODCn 11	PODCn 10	PODCn 9	PODCn 8	PODCn 7	PODCn 6	PODCn 5	PODCn 4	PODCn 3	PODCn 2	PODCn 1	PODCn 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-23 PODCn register contents

Bit position	Bit name	Function
31 to 0	PODCn[31:0]	Specifies the output buffer function: 0: Push-pull 1: Open-drain

(5) PISn - Port input buffer selection register

This register is used to specify the input buffer characteristics.

If a port has up to four input buffer characteristics, the port input buffer selection expansion register (PISEn) is also valid.

If a port has up to five input buffer characteristics, the port input buffer selection expansion register (PISEn) is also valid.

Access These registers can be read or written in 16-bit units.

Address <PORT_base> + 4700_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PISn15	PISn14	PISn13	PISn12	PISn11	PISn10	PISn9	PISn8	PISn7	PISn6	PISn5	PISn4	PISn3	PISn2	PISn1	PISn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-24 PISn register contents

Bit position	Bit name	Function
15 to 0	PISn[15:0]	Specifies the input buffer characteristic: 0: Type 1 (CMOS) 1: Type 2 (SHMT2)

Note The definition of type 1 and type 2 is given in *2.4 "V850E2/Sx4-H Port Group Configuration"*. For details about input buffer characteristics, see also the Data Sheet.

(6) PISEn - Port input buffer selection expansion register

This register is used to specify the input buffer characteristics together with the port input buffer selection register PISn.

Access This register can be read or written in 16-bit units.

Address <PORT_base> + 4800_H + n x 4

Initial value See *2.4 "V850E2/Sx4-H Port Group Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PISEn15	PISEn14	PISEn13	PISEn12	PISEn11	PISEn10	PISEn9	PISEn8	PISEn7	PISEn6	PISEn5	PISEn4	PISEn3	PISEn2	PISEn1	PISEn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-25 PISEn register contents

Bit position	Bit name	Function
15 to 0	PISEn[15:0]	These bits specify the input buffer characteristic of pin m (m = 0 to 15) together with bits PISn[15:0].

PISEnm	PISnm	Input buffer characteristic
0	0	Type 1 (CMOS)
0	1	Type 2 (SHMT2)
1	0	Type 3 (SHMT1)
1	1	Type 4 (SHMT4)

Note The definition of types 1 to 4 is given in *2.4 "V850E2/Sx4-H Port Group Configuration"*. For details about input buffer characteristics, see also the Data Sheet.

2.3.5 Port register protection

(1) PPCMDn - Port register protection command register

This register is used to specify the command to enable writing to protected port registers.

Access This register can be written in 8-bit units.

Bits 7 to 0 always return 0 when read.

Address <PORT_base> + $4C00_H$ + n x 4

Initial value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W

Table 2-26 PPCMDm register contents

Bit position	Bit name	Function
7 to 0	–	These bits specify the command that enables protected port registers to be written.

(2) PPROTSn - Port protection status register

This register shows the status of the sequence used to write data to protected port registers.

Access This register can be read in 8-bit units.

Write operation is ignored.

Address <PORT_base> + $4B00_H$ + n x 4

Initial value 00_H. This register is initialized by any reset.

<R>

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PPROTSn_0
R	R	R	R	R	R	R	R

Table 2-27 PPROTSn register contents

Bit position	Bit name	Function
0	PPROTSn_0	This bit indicates whether an error occurred in the sequence used to write data to protected port registers. 0: No protection error occurred 1: Protection error occurred

(3) Protection target port registers

PDSCn - Port drive strength control register

PODCn - Port open drain control register

(4) Write sequence used to write data to protected port registers

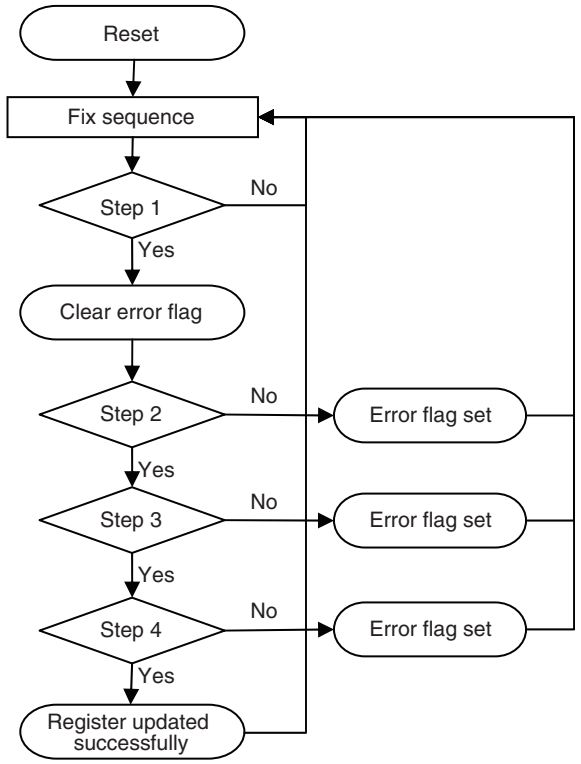


Figure 2-2 Write sequence used to write data to protected port registers

Step 1 To initialize the write sequence, write $A5_H$ to the PPCMD register.

Step 2 Write data to the protected register in 32-bit units (the register will not be updated).

Step 3 Write the inverted data to the same protected register in 32-bit units (the register will not be updated).

Step 4 Write the data again to the same protected register in 32-bit units (the register will now be updated).

2.3.6 Example port settings

Examples of the port settings are shown in the flowchart below.

Caution If the port has been set to alternative output mode by setting PIPCn.PIPCnm to 0, the port might briefly enter alternative input mode. This will occur between when the PMCn.PMCnm bit is set to 1 and when the PMn.PMnm bit is set to 0. If an interrupt-related signal is specified for a port's alternative function during this brief period, the interrupt does not occur or, even if it occurs, it is ignored completely.

(1) Batch setting

An example of specifying batch port settings is shown in the flowchart below.

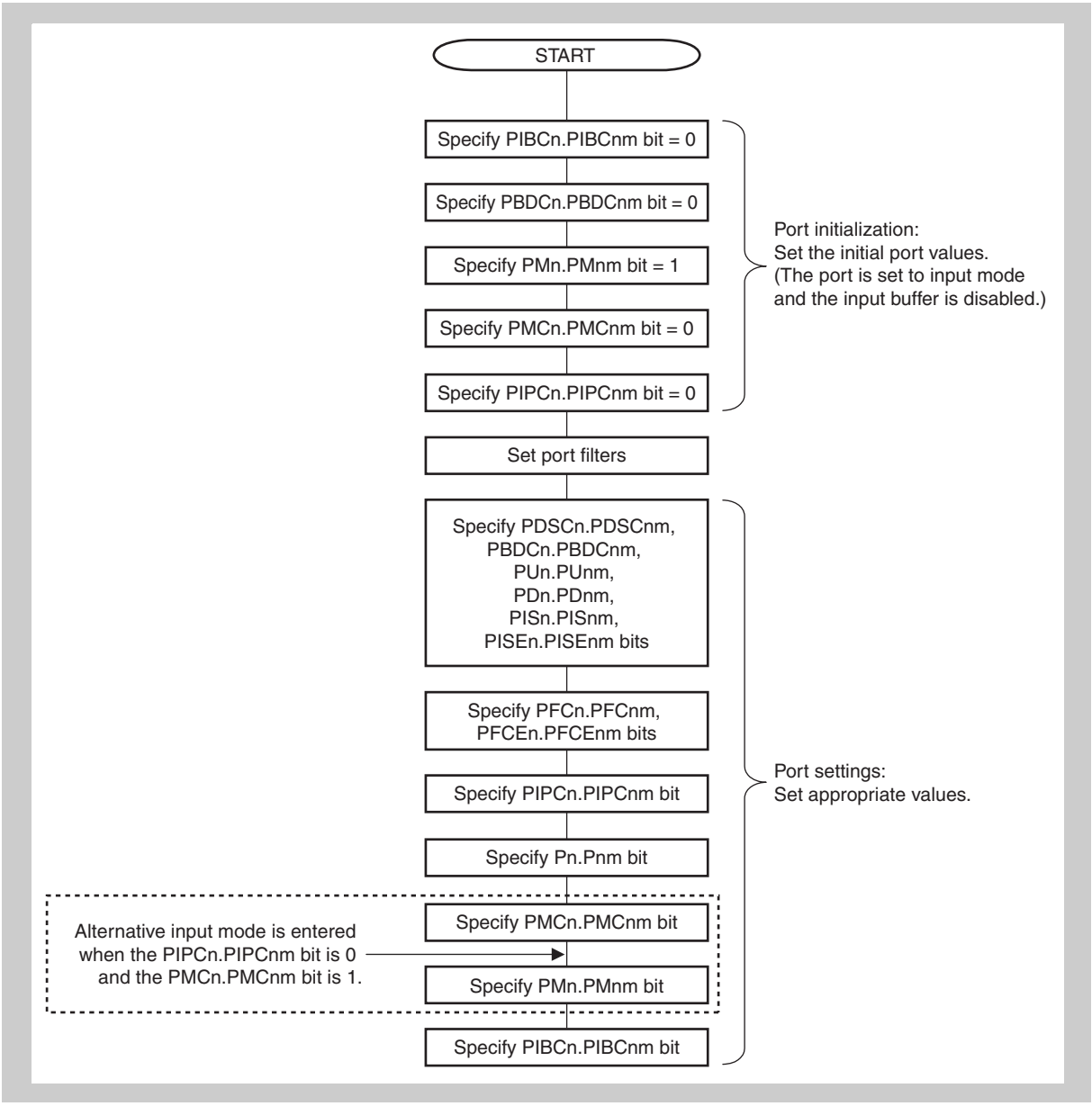


Figure 2-3 Example of port settings (when specified in batch)

(2) Individual settings

An example of specifying individual port settings is shown in the flowchart below

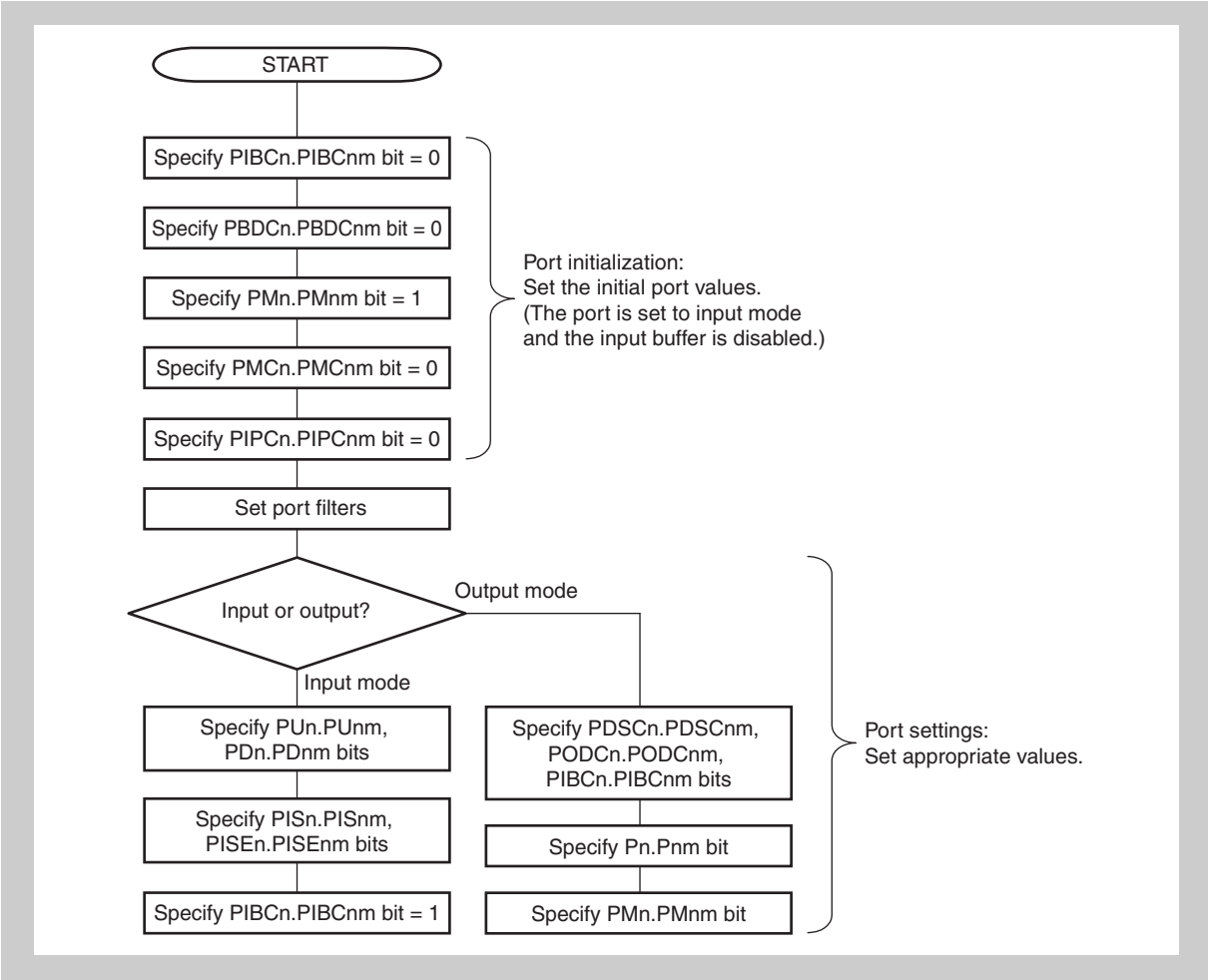


Figure 2-4 Example of port settings (in port mode)

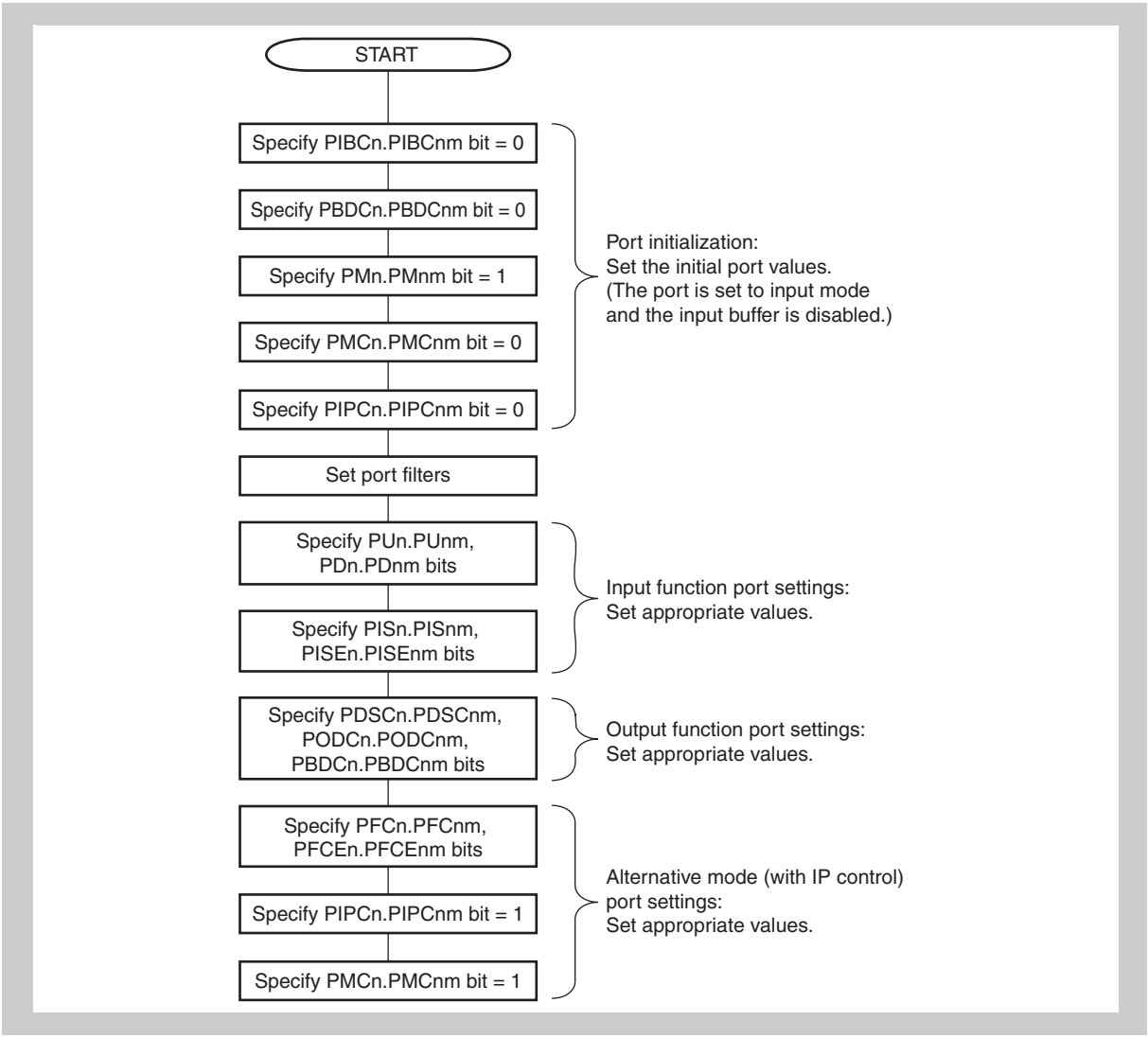


Figure 2-5 Example of port settings (in alternative mode, with IP control)

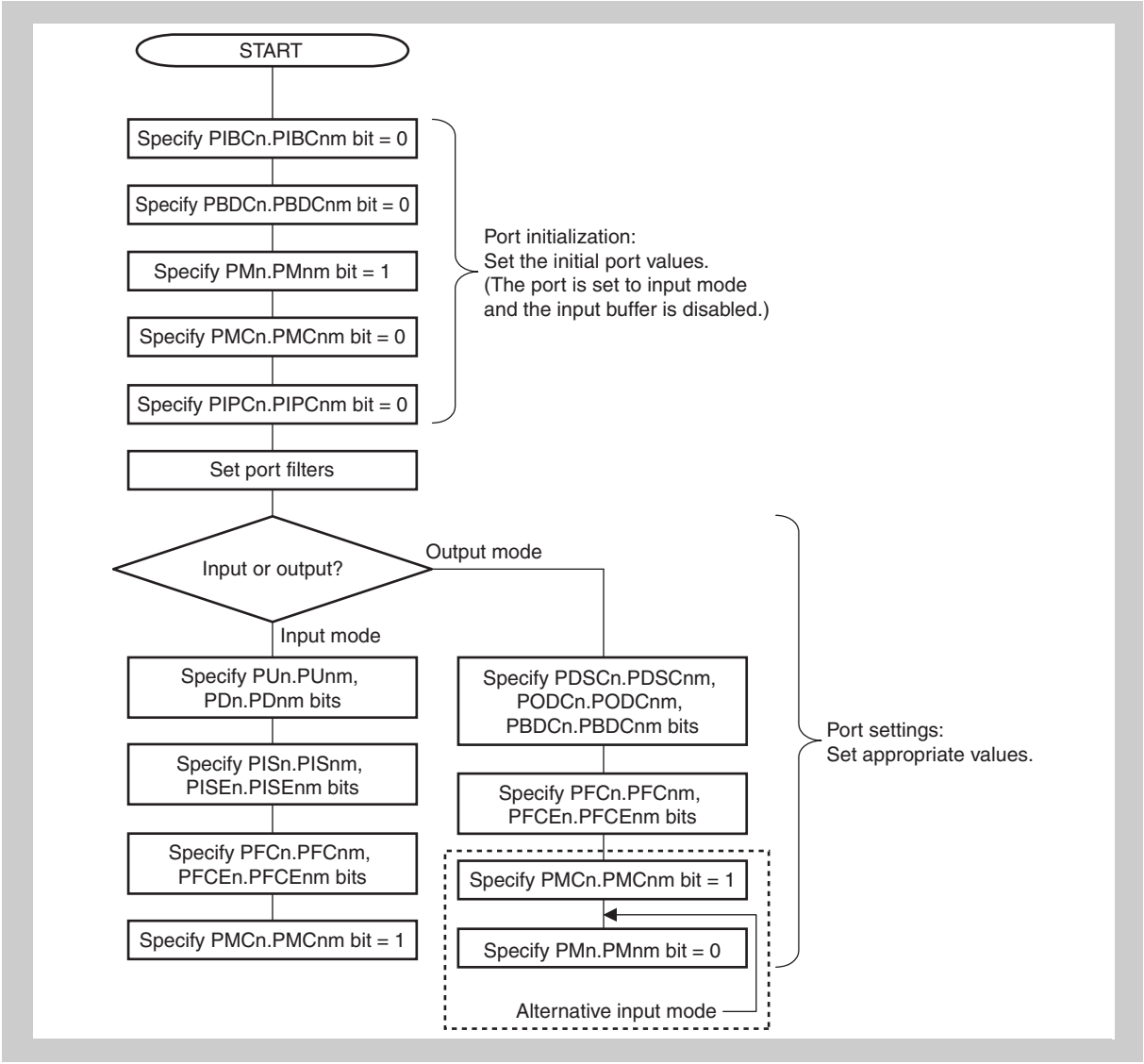


Figure 2-6 Example of port settings (in alternative mode, without IP control)

2.4 V850E2/Sx4-H Port Group Configuration

Caution The V850E2/Sx4-H port group configuration described in this section is provisional and subject to change.

This section describes the following:

- An outline of port register protection. For details, see *2.4.1 "Port register protection"*.
- A basic description of the port features. For details, see *2.4.2 "Overview of port features"*.
- The port groups in all products and their related control registers. For details, see *2.4.3 "V850E2/SG4-H port features"*.
- The alternative functions of this microcontroller and the ports to which they are assigned are listed in alphabetical order in *2.4.6 "List of pin functions in alphabetical order"*.
- For details about the statuses of ports during and after reset and after entering or exiting standby mode, see *2.4.7 "Port statuses during and after reset and after entering or exiting standby mode"*.

2.4.1 Port register protection

Some registers have port register protection.

Table 2-28 Port protection groups

Port protection groups	Port group
1	JP0
2	P0
3	P1, P3, P10
4	P21, P24 to P28

For how to write to write-protected registers, see *3.10 "Write-Protected Registers"*.

2.4.2 Overview of port features

This section describes the features of the pins of each device and their assigned alternative functions.

General-purpose I/O operation Whether the port operates in port mode or alternative mode can be selected by using the PMCn register. By setting the PMCn.PMCn_m bit to 1, the alternative function can be selected by using the PFCn and PFCEn registers. For a list of general-purpose I/O operations, see the product lists.

Special I/O after reset The special port features that operate after a reset is cancelled are described below.

(1) P0_0: $\overline{\text{RESETOUT}}$

<R> After reset is cancelled, the P0_0 pin outputs an active $\overline{\text{RESETOUT}}$ signal. The P0_0 pin operates as follows after reset is cancelled:

- PM0.PM0_0 = 0: Output port
- PODC0.PODC0_0 = 1: Open-drain output

After reset is cancelled, the P0.P0_0 bit becomes 0 and the P0_0 pin outputs an active $\overline{\text{RESETOUT}}$ signal.

If the P0_0 pin configuration is changed, output of the $\overline{\text{RESETOUT}}$ signal stops.

<R> **Caution** Once asserted, the $\overline{\text{RESETOUT}}$ signal remains low level. After reset is cancelled, therefore, the P0_0 port setting must be changed to deassert the $\overline{\text{RESETOUT}}$ signal.

(2) JP_0 to JP_5: Debug interface

After reset is cancelled, if the debug reset pin ($\overline{\text{DCUTRST}}$) is high level, the pins in the JP0 port group can be used as the debugging interface.

- JP0_0: DCUTDI input
- JP0_1: DCUTDO output
- JP0_2: DCUTCK input
- JP0_3: DCUTMS
- JP0_4: $\overline{\text{DCUTRST}}$
- JP0_5: $\overline{\text{DCURDY}}$

Therefore, while the microcontroller is connected to a debugger, these pins cannot be used as port pins or alternative function pins.

Note When connecting the JP0 to JP5 pins to a debugger, be sure to set the flash mask option OPBT0.OPBT0[31] to 1.

(3) JP0_0 to JP0_2: Flash programmer

These pins are used when the microcontroller is connected to a flash programmer.

For details, see *Chapter 7 “Flash Memory”* on page 365.

(4) Mode pins

The mode can be set by setting the following pin to 1 in combination with the FLMD0 pin:

- P0_1: FLMD1

Permanent inputs The following port pins are permanently connected to dedicated alternative functions.

Table 2-29 Permanent input connection pins

Port	Permanent inputs
P10_0	ADCA0I0
P10_1	ADCA0I1
P10_2	ADCA0I2
P10_3	ADCA0I3
P10_4	ADCA0I4
P10_5	ADCA0I5
P10_6	ADCA0I6
P10_7	ADCA0I7
P10_8	ADCA0I8
P10_9	ADCA0I9
P10_10	ADCA0I10
P10_11	ADCA0I11
P10_12	ADCA0I12
P10_13	ADCA0I13
P10_14	ADCA0I14
P10_15	ADCA0I15

Notes

1. To use the pins of port group 10 (P10) as digital inputs, the ADCA0CTL1.ADCA0GPS bit must first be set to 1.

<R>

2. The pins of port group 10 (P10) can also be used as port input/output pins. Note, however, that if the I/O level of the adjacent pin is changed or the current fluctuates due to the external circuit connected to the port pin during A/D conversion, the correct A/D conversion value might not be obtained.

Special I/O control (PIPC) Port input and output is controlled automatically by a number of functions.

Alternate functions can be specified by setting the PMCn.PMCn_m, PFCn.PFCn_m, and PFCEn.PFCEn_m bits, and I/O control can be handed over to these alternate functions by specifying the following setting:

PIPCn.PIPCn_m bit = 1

This setting disables each pin's PMn.PMn_m setting.

Alternate functions that require the PIPCn.PIPCn_m bit to be set to 1 are listed below.

Note that not all alternate functions can be used in all products.

Table 2-30 Alternate functions that require the PIPCn.PIPCn_m bit to be set to 1 (1/3)

Port	Function	Alternative mode
Clocked serial interface G (CSIG)		
P0_1	CSIG0SO	ALT_OUT3
P0_3	CSIG0SC	ALT_IN3/ALT_OUT3
P21_5	CSIG0SO	ALT_OUT3
P21_7	CSIG0SC	ALT_IN3/ALT_OUT3
P1_2	CSIG4SO	ALT_OUT3
P1_4	CSIG4SC	ALT_IN3/ALT_OUT3
P21_13	CSIG4SO	ALT_OUT3
P21_15	CSIG4SC	ALT_IN3/ALT_OUT3
Clocked serial interface H (CSIH)		
P1_7	CSIH0SO	ALT_OUT4
P1_9	CSIH0SC	ALT_IN4/ALT_OUT4
P25_2	CSIH0SO	ALT_OUT4
P25_4	CSIH0SC	ALT_IN4/ALT_OUT4
P0_5	CSIH1SO	ALT_OUT3
P0_7	CSIH1SC	ALT_IN3/ALT_OUT3
P25_10	CSIH1SO	ALT_OUT4
P25_12	CSIH1SC	ALT_IN4/ALT_OUT4
P26_2	CSIH2SO	ALT_OUT4
P26_4	CSIH2SC	ALT_IN4/ALT_OUT4
Ethernet controller (ETHA)		
P1_12	ETH0MDO	ALT_OUT4
P24_12	ETH0MDO	ALT_OUT4
External memory controller (MEMC)		
P21_0	$\overline{\text{MEMC0BEN3}}$	ALT_OUT1
	MEMC0DQM3	ALT_OUT2
P21_1	$\overline{\text{MEMC0BEN2}}$	ALT_OUT1
	MEMC0DQM2	ALT_OUT2
P21_2	$\overline{\text{MEMC0BEN1}}$	ALT_OUT1
	MEMC0DQM1	ALT_OUT2
P21_3	$\overline{\text{MEMC0BEN0}}$	ALT_OUT1
	MEMC0DQM0	ALT_OUT2
P21_4	$\overline{\text{MEMC0WR}}$	ALT_OUT1
	$\overline{\text{MEMC0WE}}$	ALT_OUT3
P21_5	$\overline{\text{MEMC0RD}}$	ALT_OUT1
P21_8	$\overline{\text{MEMC0SDRAS}}$	ALT_OUT2
P21_9	$\overline{\text{MEMC0SDCAS}}$	ALT_OUT2
P21_13	$\overline{\text{MEMC0DSTB}}$	ALT_OUT1
	MEMC0CKE	ALT_OUT2
P24_0	MEMC0AD16	ALT_IN1/ALT_OUT1
P24_1	MEMC0AD17	ALT_IN1/ALT_OUT1

Table 2-30 Alternate functions that require the PIPCn.PIPCn_m bit to be set to 1 (2/3)

Port	Function	Alternative mode
P24_2	MEMC0AD18	ALT_IN1/ALT_OUT1
P24_3	MEMC0AD19	ALT_IN1/ALT_OUT1
P24_4	MEMC0AD20	ALT_IN1/ALT_OUT1
P24_5	MEMC0AD21	ALT_IN1/ALT_OUT1
P24_6	MEMC0AD22	ALT_IN1/ALT_OUT1
P24_7	MEMC0AD23	ALT_IN1/ALT_OUT1
P24_8	MEMC0AD24	ALT_IN1/ALT_OUT1
P24_9	MEMC0AD25	ALT_IN1/ALT_OUT1
P24_10	MEMC0AD26	ALT_IN1/ALT_OUT1
P24_11	MEMC0AD27	ALT_IN1/ALT_OUT1
P24_12	MEMC0D28	ALT_IN1/ALT_OUT1
P24_13	MEMC0D29	ALT_IN1/ALT_OUT1
P24_14	MEMC0D30	ALT_IN1/ALT_OUT1
P24_15	MEMC0D31	ALT_IN1/ALT_OUT1
P25_0	MEMC0AD0	ALT_IN1/ALT_OUT1
P25_1	MEMC0AD1	ALT_IN1/ALT_OUT1
P25_2	MEMC0AD2	ALT_IN1/ALT_OUT1
P25_3	MEMC0AD3	ALT_IN1/ALT_OUT1
P25_4	MEMC0AD4	ALT_IN1/ALT_OUT1
P25_5	MEMC0AD5	ALT_IN1/ALT_OUT1
P25_6	MEMC0AD6	ALT_IN1/ALT_OUT1
P25_7	MEMC0AD7	ALT_IN1/ALT_OUT1
P25_8	MEMC0AD8	ALT_IN1/ALT_OUT1
P25_9	MEMC0AD9	ALT_IN1/ALT_OUT1
P25_10	MEMC0AD10	ALT_IN1/ALT_OUT1
P25_11	MEMC0AD11	ALT_IN1/ALT_OUT1
P25_12	MEMC0AD12	ALT_IN1/ALT_OUT1
P25_13	MEMC0AD13	ALT_IN1/ALT_OUT1
P25_14	MEMC0AD14	ALT_IN1/ALT_OUT1
P25_15	MEMC0AD15	ALT_IN1/ALT_OUT1
P26_0	MEMC0A0	ALT_OUT1
P26_1	MEMC0A1	ALT_OUT1
P26_2	MEMC0A2	ALT_OUT1
P26_3	MEMC0A3	ALT_OUT1
P26_4	MEMC0A4	ALT_OUT1
P26_5	MEMC0A5	ALT_OUT1
P26_6	MEMC0A6	ALT_OUT1
P26_7	MEMC0A7	ALT_OUT1
P26_8	MEMC0A8	ALT_OUT1
P26_9	MEMC0A9	ALT_OUT1
P26_10	MEMC0A10	ALT_OUT1

Table 2-30 Alternate functions that require the PIPCn.PIPCn_m bit to be set to 1 (3/3)

Port	Function	Alternative mode
P26_11	MEMC0A11	ALT_OUT1
P26_12	MEMC0A12	ALT_OUT1
P26_13	MEMC0A13	ALT_OUT1
P26_14	MEMC0A14	ALT_OUT1
P26_15	MEMC0A15	ALT_OUT1
P27_0	MEMC0A16	ALT_OUT1
P27_1	MEMC0A17	ALT_OUT1
P27_2	MEMC0A18	ALT_OUT1
P27_3	MEMC0A19	ALT_OUT1
P27_4	MEMC0A20	ALT_OUT1
P27_5	MEMC0A21	ALT_OUT1
P28_0	MEMC0A22	ALT_OUT1
	MEMC0A24	ALT_OUT2
P28_1	MEMC0A23	ALT_OUT1
	MEMC0A25	ALT_OUT2
PCM interface (PCM)		
P1_13	PCM0CLK	ALT_IN2/ALT_OUT2
P1_14	PCM0SEN	ALT_IN2/ALT_OUT2
P3_0	PCM1CLK	ALT_IN2/ALT_OUT2
P3_1	PCM1SEN	ALT_IN2/ALT_OUT2

2.4.3 V850E2/SG4-H port features

This section describes the port functions and alternative functions of the V850E2/SG4-H and the registers used to control the ports.

(1) General-purpose I/O operation

The port functions and alternative functions of the V850E2/SG4-H are shown in *Table 2-31 "V850E2/SG4-H general-purpose I/O operations"*.

The mode can be selected by specifying settings for the PMCn_m, PFCn_m, PFCEn_m, and PMn_m bits.

Table 2-31 V850E2/SG4-H general-purpose I/O operations (1/3)

	Port mode		Alternative mode							
	PMCn_m = 0		PMCn_m = 1							
	Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
			PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
			ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
	Port group 0 (Always-On area, E0VDD/E0VSS power supply):									
<R>	P0_0	T.B.D.	TAUJ0I0	TAUJ0O0	INTP4		$\overline{\text{CSIG0SSI}}$			IICB0SDA[b]
<R>	P0_1		TAUJ0I1	TAUJ0O1	INTP5			CSIG0SO[a]		IICB0SCL[b]
<R>	P0_2		TAUJ0I2	TAUJ0O2	INTP6		CSIG0SI			IICB1SDA[b]
<R>	P0_3		TAUJ0I3	TAUJ0O3	INTP7		CSIG0SC[a]			IICB1SCL[b]
<R>	P0_4		$\overline{\text{IEBB0RX}}$	FCN0TX	INTP8		CSIH1SI	URTE0TX		IICB2SDA[b]
<R>	P0_5		FCN0RX	$\overline{\text{IEBB0TX}}$	INTP9		URTE0RX	CSIH1SO[a]		IICB2SCL[b]
	Port group 1 (Isolated area 0, E1VDD/E1VSS power supply):									
	P1_0	T.B.D.	TAUA0I0	TAUA0O0	IISAACK		URTE1RX			CSIH0CSS0
	P1_1		TAUA0I1	TAUA0O1	IISA0SCK		$\overline{\text{CSIG4SSI}}$	URTE1TX		CSIH0CSS1
	P1_2		TAUA0I2	TAUA0O2	IISA0WS			CSIG4SO[a]		CSIH0CSS2
	P1_3		TAUA0I3	TAUA0O3	IISA0SDI	IISA0SDO	CSIG4SI			CSIH0CSS3
	P1_5		TAUA0I5	TAUA0O5	IISA1SCK				$\overline{\text{CSIH0SSI}}$	
	P1_6		TAUA0I6	TAUA0O6	IISA1WS				CSIH0SI	
	P1_7		TAUA0I7	TAUA0O7	IISA1SDI	IISA1SDO				CSIH0SO[a]
	P1_9		TAUA0I9	TAUA0O9	IISA2SCK				CSIH0SC[a]	
<R>	P1_13		TAUA0I13	TAUA0O13	PCM0CLK[a]					IICB1SCL[b]
<R>	P1_14		TAUA0I14	TAUA0O14	PCM0SEN[a]			URTE3TX		IICB3SDA[b]
<R>	P1_15		TAUA0I15	TAUA0O15	PCM0SI	PCM0SO	URTE3RX			IICB3SCL[b]
	Port group 10 (Isolated area 0, A0VDD/A0VSS power supply):									
	P10_0	T.B.D.								
	P10_1									
	P10_2									
	P10_3									
<R>	P10_4									
<R>	P10_5									
<R>	P10_6									
<R>	P10_7									

Table 2-31 V850E2/SG4-H general-purpose I/O operations (2/3)

	Port mode		Alternative mode							
	PMCn_m = 0		PMCn_m = 1							
	Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
			PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
			ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
	Port group 21 (Isolated area 1, B0VDD/B0VSS power supply):									
	P21_2	T.B.D.		$\overline{\text{MEMC0BEN1}}$[a]						
	P21_3			$\overline{\text{MEMC0BEN0}}$[a]						
<R>	P21_4			$\overline{\text{MEMC0WR}}$[a]	$\overline{\text{IEBB0RX}}$	URTE2TX	$\overline{\text{CSIG0SSI}}$	$\overline{\text{MEMC0WE}}$[a]		IICB0SDA[b]
<R>	P21_5			$\overline{\text{MEMC0RD}}$[a]	URTE2RX	$\overline{\text{IEBB0TX}}$		CSIG0SO[a]		IICB0SCL[b]
<R>	P21_6			MEMC0CLK		URTE1TX	CSIG0SI			IICB1SDA[b]
<R>	P21_7		$\overline{\text{MEMC0WAIT}}$		URTE1RX		CSIG0SC[a]			IICB1SCL[b]
	P21_15			$\overline{\text{MEMC0ASTB}}$		URTE0TX	CSIG4SC[a]			
	Port group 25 (Isolated area 1, B0VDD/B0VSS power supply):									
	P25_0	T.B.D.	MEMC0AD0[a]		TAUA0I0	TAUA0O0	IISAACK	IISA0SDO	$\overline{\text{CSIH0SSI}}$	
	P25_1		MEMC0AD1[a]		TAUA0I1	TAUA0O1	IISA0SCK		CSIH0SI	
	P25_2		MEMC0AD2[a]		TAUA0I2	TAUA0O2	IISA0WS			CSIH0SO[a]
	P25_3		MEMC0AD3[a]		TAUA0I3	TAUA0O3	IISA0SDI	IISA0SDO	CSIH0RYI	CSIH0RYO
	P25_4		MEMC0AD4[a]		TAUA0I4	TAUA0O4		IISA1SDO	CSIH0SC[a]	
	P25_5		MEMC0AD5[a]		TAUA0I5	TAUA0O5	IISA1SCK			
	P25_6		MEMC0AD6[a]		TAUA0I6	TAUA0O6	IISA1WS			
	P25_7		MEMC0AD7[a]		TAUA0I7	TAUA0O7	IISA1SDI	IISA1SDO		
	P25_8		MEMC0AD8[a]		TAUA0I8	TAUA0O8		IISA2SDO	$\overline{\text{CSIH1SSI}}$	
	P25_9		MEMC0AD9[a]		TAUA0I9	TAUA0O9	IISA2SCK		CSIH1SI	
	P25_10		MEMC0AD10[a]		TAUA0I10	TAUA0O10	IISA2WS			CSIH1SO[a]
	P25_11		MEMC0AD11[a]		TAUA0I11	TAUA0O11	IISA2SDI	IISA2SDO	CSIH1RYI	CSIH1RYO
	P25_12		MEMC0AD12[a]		TAUA0I12	TAUA0O12		IISA3SDO	CSIH1SC[a]	
	P25_13		MEMC0AD13[a]		TAUA0I13	TAUA0O13	IISA3SCK			
	P25_14		MEMC0AD14[a]		TAUA0I14	TAUA0O14	IISA3WS			
	P25_15		MEMC0AD15[a]		TAUA0I15	TAUA0O15	IISA3SDI	IISA3SDO		
<R>	Port group 27 (Isolated area 1, B0VDD/B0VSS power supply):									
	P27_0	T.B.D.	INTP0	MEMC0A16[a]			IISA3SDI	IISA3SDO		CSIH1CSS0
	P27_1		INTP1	MEMC0A17[a]			IISA2SDI	IISA2SDO		CSIH1CSS1
	P27_2		INTP2	MEMC0A18[a]			IISA1SDI	IISA1SDO		CSIH1CSS2
	P27_3		INTP3	MEMC0A19[a]			IISA0SDI	IISA0SDO		CSIH1CSS3
	P27_4		INTP4	MEMC0A20[a]			IISA0SCK			URTE10TX
	P27_5		INTP5	MEMC0A21[a]			IISA0WS		URTE10RX	
	Port group JP0 (Always-On area, E0VDD/E0VSS power supply):[c]									
	JP0_0	T.B.D.	INTP0		TAUJ0I0	TAUJ0O0				
	JP0_1		INTP1		TAUJ0I1	TAUJ0O1				
	JP0_2		INTP2		TAUJ0I2	TAUJ0O2				
	JP0_3		INTP3		TAUJ0I3	TAUJ0O3				

Table 2-31 V850E2/SG4-H general-purpose I/O operations (3/3)

Port mode		Alternative mode							
PMCn_m = 0		PMCn_m = 1							
Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
		PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
		ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
JP0_4	T.B.D.								
JP0_5		NMI	RTCA0OUT						

a) The PIPCn.PIPCn_m bit must be set to 1 to use this alternative mode. The mode can then be set to I/O, after which the PMn.PMn_m bit setting is ignored.

<R> b) When using the following alternate-function pins, set the PBDCn.PBDCnm bit to 1.

c) JP0_0 to JP0_5 are used for debugging. For details, see *2.4.2 “Overview of port features” on page 76*.

(2) V850E2/SG4-H port control registers

This section describes the registers used to control the ports in the V850E2/SG4-H and their addresses and initial values.

Key for symbols in tables

A: Register address

I: Initial value

B: Valid bit

- 1: Valid, X: Invalid
- Right side: Bit 0, left side: Bit 15

Table 2-32 V850E2/SG4-H port control registers (groups 0, 1, 10, 21) (1/2)

	Register		Port group n =			
			0	1	10	21
	Pn	A:	FF40 0000$_H$	FF40 0004$_H$	FF40 0028$_H$	FF40 0054$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111	xxxx xxxx 1111 1111	1xxx xxxx 1111 11xx
	PSRn	A:	FF40 0100$_H$	FF40 0104$_H$	FF40 0128$_H$	FF40 0154$_H$
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111	xxxx xxxx 1111 1111	1xxx xxxx 1111 11xx
	PNOTn	A:	FF40 0700$_H$	FF40 0704$_H$	FF40 0728$_H$	FF40 0754$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111	xxxx xxxx 1111 1111	1xxx xxxx 1111 11xx
	PPRn	A:	FF40 0200$_H$	FF40 0204$_H$	FF40 0228$_H$	FF40 0254$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111	xxxx xxxx 1111 1111	1xxx xxxx 1111 11xx
	PMn	A:	FF40 0300$_H$	FF40 0304$_H$	FF40 0328$_H$	FF40 0354$_H$
		I:	FFFE$_H$	FFFF$_H$	FFFF$_H$	FFFF$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111	xxxx xxxx 1111 1111	1xxx xxxx 1111 11xx
	PMCn	A:	FF40 0400$_H$	FF40 0404$_H$	–	FF40 0454$_H$
		I:	0000$_H$	0000$_H$		0000$_H$
		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PFCn	A:	FF40 0500$_H$	FF40 0504$_H$	–	FF40 0554$_H$
		I:	0000$_H$	0000$_H$		0000$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PFCEn	A:	FF40 0600$_H$	FF40 0604$_H$	–	FF40 0654$_H$
		I:	0000$_H$	0000$_H$		0000$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PMSRn	A:	FF40 0800$_H$	FF40 0804$_H$	FF40 0828$_H$	FF40 0854$_H$
		I:	0000 0000 0000 FFFE$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111	xxxx xxxx 1111 1111	1xxx xxxx 1111 11xx

Table 2-32 V850E2/SG4-H port control registers (groups 0, 1, 10, 21) (2/2)

	Register		Port group n =			
			0	1	10	21
	PMCSRn	A:	FF40 0900$_H$	FF40 0904$_H$	–	FF40 0954$_H$
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$		0000 0000 0000 0000$_H$
		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PIBCn	A:	FF40 4000$_H$	FF40 4004$_H$	FF40 4028$_H$	FF40 4054$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111	xxxx xxxx 1111 1111	1xxx xxxx 1111 11xx
	PBDCn	A:	FF40 4100$_H$	FF40 4104$_H$	FF40 4128$_H$	FF40 4154$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111	xxxx xxxx 1111 1111	1xxx xxxx 1111 11xx
	PIPCn	A:	FF40 4200$_H$	FF40 4204$_H$	–	FF40 4254$_H$
		I:	0000$_H$	0000$_H$		0000$_H$
		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PUn	A:	FF40 4300$_H$	FF40 4304$_H$	–	FF40 4354$_H$
		I:	0000$_H$	0000$_H$		0000$_H$
		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PDn	A:	FF40 4400$_H$	FF40 4404$_H$	–	FF40 4454$_H$
		I:	0000$_H$	0000$_H$		0000$_H$
		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PODCn	A:	FF40 4500$_H$	FF40 4504$_H$	FF40 4528$_H$	FF40 4554$_H$
		I:	0000 0000 0000 0001$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
<R>		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111	xxxx xxxx 1111 1111	1xxx xxxx 1111 11xx
	PDSCn	A:	FF40 4600$_H$	FF40 4604$_H$	–	FF40 4654$_H$
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$		0000 0000 0000 0000$_H$
		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PISn	A:	FF40 4700$_H$	FF40 4704$_H$	–	FF40 4754$_H$
		I:	0000$_H$	0000$_H$		0000$_H$
		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PISEn	A:	FF40 4800$_H$	FF40 4804$_H$	–	FF40 4854$_H$
		I:	0000$_H$	0000$_H$		0000$_H$
		B:	xxxx xxxx xx11 1111	111x xx1x 111x 1111		1xxx xxxx 1111 11xx
	PPCMDn	A:	FF40 4C00$_H$	FF40 4C04$_H$	FF40 4C28$_H$	FF40 4C54$_H$
		I:	00$_H$	00$_H$	00$_H$	00$_H$
		B:	1111 1111	1111 1111	1111 1111	1111 1111
	PPROTSn	A:	FF40 4B00$_H$	FF40 4B04$_H$	FF40 4B28$_H$	FF40 4B54$_H$
		I:	00$_H$	00$_H$	00$_H$	00$_H$
		B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-33 V850E2/SG4-H port control registers (groups 25, 27, JP0) (1/2)

	Register		Port group n =			
			25	27	JP0	
	Pn	A:	FF40 0064$_H$	FF40 006C$_H$	FF44 0000$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PSRn	A:	FF40 0164$_H$	FF40 016C$_H$	FF44 0010$_H$	
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xxxx xxxx xx11 1111	
	PNOTn	A:	FF40 0764$_H$	FF40 076C$_H$	FF44 0070$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PPRn	A:	FF40 0264$_H$	FF40 026C$_H$	FF44 0020$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PMn	A:	FF40 0364$_H$	FF40 036C$_H$	FF44 0030$_H$	
		I:	FFFF$_H$	FFFF$_H$	FF$_H$.	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PMCn	A:	FF40 0464$_H$	FF40 046C$_H$	FF44 0040$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx1x 1111	
	PFCn	A:	FF40 0564$_H$	FF40 056C$_H$	FF44 0050$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xxxx 1111	
	PFCEn	A:	FF40 0664$_H$	FF40 066C$_H$	–	
		I:	0000$_H$	0000$_H$		
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111		
	PMSRn	A:	FF40 0864$_H$	FF40 086C$_H$	FF44 0080$_H$	
		I:	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 00FF$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xxxx xxxx xx11 1111	
	PMCSRn	A:	FF40 0964$_H$	FF40 096C$_H$	FF44 0090$_H$	
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xxxx xxxx xx1x 1111	
	PIBCn	A:	FF40 4064$_H$	FF40 406C$_H$	FF44 0400$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PBDCn	A:	FF40 4164$_H$	FF40 416C$_H$	FF44 0410$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PIPCn	A:	FF40 4264$_H$	FF40 426C$_H$	FF44 0420$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	

Table 2-33 V850E2/SG4-H port control registers (groups 25, 27, JP0) (2/2)

	Register		Port group n =			
			25	27	JP0	
	PUn	A:	FF40 4364$_H$	FF40 436C$_H$	FF44 0430$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PDn	A:	FF40 4464$_H$	FF40 446C$_H$	FF44 0440$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PODCn	A:	FF40 4564$_H$	FF40 456C$_H$	FF44 0450$_H$	
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xxxx xxxx xx11 1111	
	PDSCn	A:	FF40 4664$_H$	FF40 466C$_H$	FF44 0460$_H$	
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xxxx xxxx xx11 1111	
	PISn	A:	FF40 4764$_H$	FF40 476C$_H$	FF44 0470$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PISEn	A:	FF40 4864$_H$	FF40 486C$_H$	FF44 0480$_H$	
		I:	0000$_H$	0000$_H$	00$_H$	
<R>		B:	1111 1111 1111 1111	xxxx xxxx xxxx 1111	xx11 1111	
	PPCMDn	A:	FF40 4C64$_H$	FF40 4C6C$_H$	FF44 04C0$_H$	
		I:	00$_H$	00$_H$	00$_H$	
<R>		B:	1111 1111	1111 1111	1111 1111	
	PPROTSn	A:	FF40 4B64$_H$	FF40 4B6C$_H$	FF44 04B0$_H$	
		I:	00$_H$	00$_H$	00$_H$	
		B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	

2.4.4 V850E2/SJ4-H port features

This section describes the port functions and alternative functions of the V850E2/SJ4-H and the registers used to control the ports.

(1) General-purpose I/O operation

The port functions and alternative functions of the V850E2/SJ4-H are shown in *Table 2-34 "V850E2/SJ4-H general-purpose I/O operations"*.

The mode can be selected by specifying settings for the PMCn_m, PFCn_m, PFCEn_m, and PMn_m bits.

Table 2-34 V850E2/SJ4-H general-purpose I/O operations (1/4)

	Port mode		Alternative mode							
	PMCn_m = 0		PMCn_m = 1							
	Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
			PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
			ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
	Port group 0 (Always-On area, E0VDD/E0VSS power supply):									
<R>	P0_0	T.B.D.	TAUJ0I0	TAUJ0O0	INTP4		$\overline{\text{CSIG0SSI}}$			IICB0SDA[b]
<R>	P0_1		TAUJ0I1	TAUJ0O1	INTP5			CSIG0SO[a]		IICB0SCL[b]
<R>	P0_2		TAUJ0I2	TAUJ0O2	INTP6		CSIG0SI			IICB1SDA[b]
<R>	P0_3		TAUJ0I3	TAUJ0O3	INTP7		CSIG0SC[a]			IICB1SCL[b]
<R>	P0_4		$\overline{\text{IEBB0RX}}$	FCN0TX	INTP8		CSIH1SI	URTE0TX		IICB2SDA[b]
<R>	P0_5		FCN0RX	$\overline{\text{IEBB0TX}}$	INTP9		URTE0RX	CSIH1SO[a]		IICB2SCL[b]
	P0_6			FCN1TX	INTP10		CSIH1RYI	CSIH1RYO	$\overline{\text{CSIH1SSI}}$	URTE10TX
	P0_7		FCN1RX		INTP11		CSIH1SC[a]		URTE10RX	
	Port group 1 (Isolated area 0, E1VDD/E1VSS power supply):									
	P1_0	T.B.D.	TAUA0I0	TAUA0O0	IISAACK		URTE1RX			CSIH0CSS0
	P1_1		TAUA0I1	TAUA0O1	IISA0SCK		$\overline{\text{CSIG4SSI}}$	URTE1TX		CSIH0CSS1
	P1_2		TAUA0I2	TAUA0O2	IISA0WS			CSIG4SO[a]		CSIH0CSS2
	P1_3		TAUA0I3	TAUA0O3	IISA0SDI	IISA0SDO	CSIG4SI			CSIH0CSS3
	P1_5		TAUA0I5	TAUA0O5	IISA1SCK				$\overline{\text{CSIH0SSI}}$	
	P1_6		TAUA0I6	TAUA0O6	IISA1WS				CSIH0SI	
	P1_7		TAUA0I7	TAUA0O7	IISA1SDI	IISA1SDO				CSIH0SO[a]
	P1_9		TAUA0I9	TAUA0O9	IISA2SCK				CSIH0SC[a]	
	P1_10		TAUA0I10	TAUA0O10	IISA2WS				URTE2RX	
	P1_11		TAUA0I11	TAUA0O11	IISA2SDI	IISA2SDO				URTE2TX
<R>	P1_13		TAUA0I13	TAUA0O13	PCM0CLK[a]					IICB1SCL[b]
<R>	P1_14		TAUA0I14	TAUA0O14	PCM0SEN[a]			URTE3TX		IICB3SDA[b]
<R>	P1_15		TAUA0I15	TAUA0O15	PCM0SI	PCM0SO	URTE3RX			IICB3SCL[b]
	Port group 3 (Isolated area 0, E1VDD/E1VSS power supply):									
<R>	P3_0	T.B.D.			PCM1CLK[a]		KR0I0	CSIH1CSS0		IICB0SDA[b]
<R>	P3_1				PCM1SEN[a]		KR0I1	CSIH1CSS1		IICB0SCL[b]
<R>	P3_2				PCM1SI	PCM1SO	KR0I2	CSIH1CSS2		IICB1SDA[b]
	P3_3				IISAACK		KR0I3	CSIH1CSS3		

Table 2-34 V850E2/SJ4-H general-purpose I/O operations (2/4)

	Port mode		Alternative mode							
	PMCn_m = 0		PMCn_m = 1							
	Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
			PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
			ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
	Port group 10 (Isolated area 0, A0VDD/A0VSS power supply):									
	P10_0:	T.B.D.								
	P10_1									
	P10_2									
	P10_3									
	P10_4									
	P10_5									
	P10_6									
	P10_7									
	P10_8									
	P10_9		ADCA0TRG0							
	P10_10		ADCA0TRG1							
	P10_11		ADCA0TRG2							
	P10_12									
	P10_13									
<R>	P10_14									
<R>	P10_15									
	Port group 21 (Isolated area 1, B0VDD/B0VSS power supply):									
	P21_2	T.B.D.		$\overline{\text{MEMC0BEN1}}$[a]		MEMC0DQM1[a]				
	P21_3			$\overline{\text{MEMC0BEN0}}$[a]		MEMC0DQM0[a]				
<R>	P21_4			$\overline{\text{MEMC0WR}}$[a]	$\overline{\text{IEBB0RX}}$	URTE2TX	$\overline{\text{CSIG0SSI}}$	$\overline{\text{MEMC0WE}}$[a]		IICB0SDA[b]
<R>	P21_5			$\overline{\text{MEMC0RD}}$[a]	URTE2RX	$\overline{\text{IEBB0TX}}$		CSIG0SO[a]		IICB0SCL[b]
<R>	P21_6			MEMC0CLK		URTE1TX	CSIG0SI			IICB1SDA[b]
<R>	P21_7		$\overline{\text{MEMC0WAIT}}$		URTE1RX		CSIG0SC[a]			IICB1SCL[b]
<R>	P21_8					$\overline{\text{MEMC0SDRAS}}$[a]				IICB3SDA[b]
<R>	P21_9			$\overline{\text{MEMC0CS2}}$		$\overline{\text{MEMC0SDCAS}}$[a]				IICB3SCL[b]
	P21_10			$\overline{\text{MEMC0CS3}}$	FCN1RX					
	P21_11			$\overline{\text{MEMC0CS4}}$		FCN1TX				
<R>	P21_13		$\overline{\text{MEMC0HLDRQ}}$	$\overline{\text{MEMC0DSTB}}$[a]	FCN0RX	MEMC0CKE[a]		CSIG4SO[a]		IICB2SDA[b]
<R>	P21_14			$\overline{\text{MEMC0HLDAK}}$	URTE0RX	FCN0TX	CSIG4SI			IICB2SCL[b]
	P21_15			$\overline{\text{MEMC0ASTB}}$		URTE0TX	CSIG4SC[a]			
	Port group 25 (Isolated area 1, B0VDD/B0VSS power supply):									
	P25_0	T.B.D.	MEMC0AD0[a]		TAUA0I0	TAUA0O0	IISAACK	IISA0SDO	$\overline{\text{CSIH0SSI}}$	
	P25_1		MEMC0AD1[a]		TAUA0I1	TAUA0O1	IISA0SCK		CSIH0SI	
	P25_2		MEMC0AD2[a]		TAUA0I2	TAUA0O2	IISA0WS			CSIH0SO[a]
	P25_3		MEMC0AD3[a]		TAUA0I3	TAUA0O3	IISA0SDI	IISA0SDO	CSIH0RYI	CSIH0RYO
	P25_4		MEMC0AD4[a]		TAUA0I4	TAUA0O4		IISA1SDO	CSIH0SC[a]	
	P25_5		MEMC0AD5[a]		TAUA0I5	TAUA0O5	IISA1SCK			

Table 2-34 V850E2/SJ4-H general-purpose I/O operations (3/4)

Port mode		Alternative mode							
PMCn_m = 0		PMCn_m = 1							
Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
		PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
		ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
P25_6	T.B.D.	MEMC0AD6[a]		TAUA0I6	TAUA0O6	IISA1WS			
P25_7		MEMC0AD7[a]		TAUA0I7	TAUA0O7	IISA1SDI	IISA1SDO		
P25_8		MEMC0AD8[a]		TAUA0I8	TAUA0O8		IISA2SDO	$\overline{\text{CSIH1SSI}}$	
P25_9		MEMC0AD9[a]		TAUA0I9	TAUA0O9	IISA2SCK		CSIH1SI	
P25_10		MEMC0AD10[a]		TAUA0I10	TAUA0O10	IISA2WS			CSIH1SO[a]
P25_11		MEMC0AD11[a]		TAUA0I11	TAUA0O11	IISA2SDI	IISA2SDO	CSIH1RYI	CSIH1RYO
P25_12		MEMC0AD12[a]		TAUA0I12	TAUA0O12		IISA3SDO	CSIH1SC[a]	
P25_13		MEMC0AD13[a]		TAUA0I13	TAUA0O13	IISA3SCK			
P25_14		MEMC0AD14[a]		TAUA0I14	TAUA0O14	IISA3WS			
P25_15		MEMC0AD15[a]		TAUA0I15	TAUA0O15	IISA3SDI	IISA3SDO		
Port group 26 (Isolated area 1, B0VDD/B0VSS power supply):									
P26_0	T.B.D.	KR0I0	MEMC0A0[a]					$\overline{\text{CSIH2SSI}}$	
P26_1		KR0I1	MEMC0A1[a]					CSIH2SI	
P26_2		KR0I2	MEMC0A2[a]						CSIH2SO[a]
P26_3		KR0I3	MEMC0A3[a]					CSIH2RYI	CSIH2RYO
P26_4		KR0I4	MEMC0A4[a]					CSIH2SC[a]	
P26_5		KR0I5	MEMC0A5[a]						CSIH2CSS0
P26_6		KR0I6	MEMC0A6[a]						CSIH2CSS1
P26_7		KR0I7	MEMC0A7[a]						CSIH2CSS2
P26_8		INTP8	MEMC0A8[a]			IISA3SCK			CSIH2CSS3
P26_9		INTP9	MEMC0A9[a]			IISA3WS			CSIH2CSS4
P26_10		INTP10	MEMC0A10[a]			IISA2SCK			CSIH2CSS5
P26_11		INTP11	MEMC0A11[a]			IISA2WS			CSIH2CSS6
P26_12		INTP12	MEMC0A12[a]			IISA0SCK			CSIH2CSS7
P26_13		INTP13	MEMC0A13[a]			IISA0WS			
P26_14		INTP14	MEMC0A14[a]						
P26_15		INTP15	MEMC0A15[a]						
Port group 27 (Isolated area 1, B0VDD/B0VSS power supply):									
P27_0	T.B.D.	INTP0	MEMC0A16[a]			IISA3SDI	IISA3SDO		CSIH1CSS0
P27_1		INTP1	MEMC0A17[a]			IISA2SDI	IISA2SDO		CSIH1CSS1
P27_2		INTP2	MEMC0A18[a]			IISA1SDI	IISA1SDO		CSIH1CSS2
P27_3		INTP3	MEMC0A19[a]			IISA0SDI	IISA0SDO		CSIH1CSS3
P27_4		INTP4	MEMC0A20[a]			IISA0SCK			URTE10TX
P27_5		INTP5	MEMC0A21[a]			IISA0WS		URTE10RX	
Port group 28 (Isolated area 1, B0VDD/B0VSS power supply):									
P28_0	T.B.D.	INTP6	MEMC0A22[a]		MEMC0A24[a]	IISA1SCK			
P28_1		INTP7	MEMC0A23[a]		MEMC0A25[a]	IISA1WS			

Table 2-34 V850E2/SJ4-H general-purpose I/O operations (4/4)

Port mode		Alternative mode							
PMCn_m = 0		PMCn_m = 1							
Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
		PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
		ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
Port group JP0 (Always-On area, E0VDD/E0VSS power supply):[c]									
JP0_0	T.B.D.	INTP0		TAUJ0I0	TAUJ0O0				
JP0_1		INTP1		TAUJ0I1	TAUJ0O1				
JP0_2		INTP2		TAUJ0I2	TAUJ0O2				
JP0_3		INTP3		TAUJ0I3	TAUJ0O3				
JP0_4									
JP0_5		NMI	RTCA0OUT						

a) The PIPCn.PIPCn_m bit must be set to 1 to use this alternative mode. The mode can then be set to I/O, after which the PMn.PMn_m bit setting is ignored.

<R> b) When using the following alternate-function pins, set the PBDCn.PBDCnm bit to 1.

c) JP0_0 to JP0_5 are used for debugging. For details, see *2.4.2 “Overview of port features” on page 76*.

(2) V850E2/SJ4-H port control registers

This section describes the registers used to control the ports in the V850E2/SJ4-H and their addresses and initial values.

Key for symbols in tables

A: Register address

I: Initial value

B: Valid bit

– 1: Valid, X: Invalid

– Right side: Bit 0, left side: Bit 15

Table 2-35 V850E2/SJ4-H port control registers (groups 0, 1, 3, 10) (1/2)

	Register		Port group n =			
			0	1	3	10
	Pn	A:	FF40 0000$_H$	FF40 0004$_H$	FF40 000C$_H$	FF40 0028$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	1111 1111 1111 1111
	PSRn	A:	FF40 0100$_H$	FF40 0104$_H$	FF40 010C$_H$	FF40 0128$_H$
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
<R>		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	1111 1111 1111 1111
	PNOTn	A:	FF40 0700$_H$	FF40 0704$_H$	FF40 070C$_H$	FF40 0728$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	1111 1111 1111 1111
	PPRn	A:	FF40 0200$_H$	FF40 0204$_H$	FF40 020C$_H$	FF40 0228$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	1111 1111 1111 1111
	PMn	A:	FF40 0300$_H$	FF40 0304$_H$	FF40 030C$_H$	FF40 0328$_H$
		I:	FFFE$_H$	FFFF$_H$	FFFF$_H$	FFFF$_H$
<R>		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	1111 1111 1111 1111
	PMCn	A:	FF40 0400$_H$	FF40 0404$_H$	FF40 040C$_H$	FF40 0428$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	xxxx 111x xxxx xxxx
	PFCn	A:	FF40 0500$_H$	FF40 0504$_H$	FF40 050C$_H$	–
		I:	0000$_H$	0000$_H$	0000$_H$	
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	
	PFCEn	A:	FF40 0600$_H$	FF40 0604$_H$	FF40 060C$_H$	–
		I:	0000$_H$	0000$_H$	0000$_H$	
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	
	PMSRn	A:	FF40 0800$_H$	FF40 0804$_H$	FF40 080C$_H$	FF40 0828$_H$
		I:	0000 0000 0000 FFFE$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$
<R>		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	1111 1111 1111 1111
	PMCSRn	A:	FF40 0900$_H$	FF40 0904$_H$	FF40 090C$_H$	FF40 0928$_H$
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	xxxx 111x xxxx xxxx

Table 2-35 V850E2/SJ4-H port control registers (groups 0, 1, 3, 10) (2/2)

	Register		Port group n =			
			0	1	3	10
	PIBCn	A:	FF40 4000$_H$	FF40 4004$_H$	FF40 400C$_H$	FF40 4028$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	1111 1111 1111 1111
	PBDCn	A:	FF40 4100$_H$	FF40 4104$_H$	FF40 410C$_H$	FF40 4128$_H$
		I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	1111 1111 1111 1111
	PIPCn	A:	FF40 4200$_H$	FF40 4204$_H$	FF40 420C$_H$	–
		I:	0000$_H$	0000$_H$	0000$_H$	
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	
	PUn	A:	FF40 4300$_H$	FF40 4304$_H$	FF40 430C$_H$	–
		I:	0000$_H$	0000$_H$	0000$_H$	
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	
	PDn	A:	FF40 4400$_H$	FF40 4404$_H$	FF40 440C$_H$	–
		I:	0000$_H$	0000$_H$	0000$_H$	
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	
	PODCn	A:	FF40 4500$_H$	FF40 4504$_H$	FF40 450C$_H$	FF40 4528$_H$
		I:	0000 0000 0000 0001$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
<R>		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	1111 1111 1111 1111
	PDSCn	A:	FF40 4600$_H$	FF40 4604$_H$	FF40 460C$_H$	–
		I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	
	PISn	A:	FF40 4700$_H$	FF40 4704$_H$	FF40 470C$_H$	–
		I:	0000$_H$	0000$_H$	0000$_H$	
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	
	PISEn	A:	FF40 4800$_H$	FF40 4804$_H$	FF40 480C$_H$	–
		I:	0000$_H$	0000$_H$	0000$_H$	
		B:	xxxx xxxx 1111 1111	111x 111x 111x 1111	xxxx xxxx xxxx 1111	
	PPCMDn	A:	FF40 4C00$_H$	FF40 4C04$_H$	FF40 4C0C$_H$	FF40 4C28$_H$
		I:	00$_H$	00$_H$	00$_H$	00$_H$
		B:	1111 1111	1111 1111	1111 1111	1111 1111
	PPROTSn	A:	FF40 4B00$_H$	FF40 4B04$_H$	FF40 4B0C$_H$	FF40 4B28$_H$
		I:	00$_H$	00$_H$	00$_H$	00$_H$
		B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-36 V850E2/SJ4-H port control registers (groups 21, 25 to 27) (1/2)

Register		Port group n =			
		21	25	26	27
Pn	A:	FF40 0054$_H$	FF40 0064$_H$	FF40 0068$_H$	FF40 006C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PSRn	A:	FF40 0154$_H$	FF40 0164$_H$	FF40 0168$_H$	FF40 016C$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PNOTn	A:	FF40 0754$_H$	FF40 0764$_H$	FF40 0768$_H$	FF40 076C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PPRn	A:	FF40 0254$_H$	FF40 0264$_H$	FF40 0268$_H$	FF40 026C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PMn	A:	FF40 0354$_H$	FF40 0364$_H$	FF40 0368$_H$	FF40 036C$_H$
	I:	FFFF$_H$	FFFF$_H$	FFFF$_H$	FFFF$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PMCn	A:	FF40 0454$_H$	FF40 0464$_H$	FF40 0468$_H$	FF40 046C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PFCn	A:	FF40 0554$_H$	FF40 0564$_H$	FF40 0568$_H$	FF40 056C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>	B:	111x 1111 1111 11xx	1111 1111 1111 1111	xxx1 1111 1111 1111	xxxx xxxx xx11 1111
PFCEn	A:	FF40 0654$_H$	FF40 0664$_H$	FF40 0668$_H$	FF40 066C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>	B:	111x xx11 1111 11xx	1111 1111 1111 1111	xx11 1111 1111 1111	xxxx xxxx xx11 1111
PMSRn	A:	FF40 0854$_H$	FF40 0864$_H$	FF40 0868$_H$	FF40 086C$_H$
	I:	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PMCSRn	A:	FF40 0954$_H$	FF40 0964$_H$	FF40 0968$_H$	FF40 096C$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PIBCn	A:	FF40 4054$_H$	FF40 4064$_H$	FF40 4068$_H$	FF40 406C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PBDCn	A:	FF40 4154$_H$	FF40 4164$_H$	FF40 4168$_H$	FF40 416C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PIPCn	A:	FF40 4254$_H$	FF40 4264$_H$	FF40 4268$_H$	FF40 426C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111

Table 2-36 V850E2/SJ4-H port control registers (groups 21, 25 to 27) (2/2)

Register		Port group n =			
		21	25	26	27
PUn	A:	FF40 4354$_H$	FF40 4364$_H$	FF40 4368$_H$	FF40 436C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PDn	A:	FF40 4454$_H$	FF40 4464$_H$	FF40 4468$_H$	FF40 446C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PODCn	A:	FF40 4554$_H$	FF40 4564$_H$	FF40 4568$_H$	FF40 456C$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PDSCn	A:	FF40 4654$_H$	FF40 4664$_H$	FF40 4668$_H$	FF40 466C$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PISn	A:	FF40 4754$_H$	FF40 4764$_H$	FF40 4768$_H$	FF40 476C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PISEn	A:	FF40 4854$_H$	FF40 4864$_H$	FF40 4868$_H$	FF40 486C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	111x 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 1111
PPCMDn	A:	FF40 4C54$_H$	FF40 4C64$_H$	FF40 4C68$_H$	FF40 4C6C$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	1111 1111	1111 1111	1111 1111	1111 1111
PPROTSn	A:	FF40 4B54$_H$	FF40 4B64$_H$	FF40 4B68$_H$	FF40 4B6C$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-37 V850E2/SJ4-H port control registers (groups 28, JP0) (1/2)

Register		Port group n =			
		28	JP0		
Pn	A:	FF40 0070$_H$	FF44 0000$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PSRn	A:	FF40 0170$_H$	FF44 0010$_H$		
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$		
	B:	xxxx xxxx xxxx xx11	xxxx xxxx xx11 1111		
PNOTn	A:	FF40 0770$_H$	FF44 0700$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PPRn	A:	FF40 0270$_H$	FF44 0020$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PMn	A:	FF40 0370$_H$	FF44 0030$_H$		
	I:	FFFF$_H$	FF$_H$.		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PMCn	A:	FF40 0470$_H$	FF44 0040$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx1x 1111		
PFCn	A:	FF40 0570$_H$	FF44 0050$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xxxx 1111		
PFCEn	A:	FF40 0670$_H$	–		
	I:	0000$_H$			
	B:	xxxx xxxx xxxx xx11			
PMSRn	A:	FF40 0870$_H$	FF44 0080$_H$		
	I:	0000 0000 0000 FFFF$_H$	0000 0000 0000 00FF$_H$		
	B:	xxxx xxxx xxxx xx11	xxxx xxxx xx11 1111		
PMCSRn	A:	FF40 0970$_H$	FF44 0090$_H$		
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$		
	B:	xxxx xxxx xxxx xx11	xxxx xxxx xx1x 1111		
PIBCn	A:	FF40 4070$_H$	FF44 0400$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PBDCn	A:	FF40 4170$_H$	FF44 0410$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PIPCn	A:	FF40 4270$_H$	FF44 0420$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		

Table 2-37 V850E2/SJ4-H port control registers (groups 28, JP0) (2/2)

Register		Port group n =			
		28	JP0		
PUn	A:	FF40 4370$_H$	FF44 0430$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PDn	A:	FF40 4470$_H$	FF44 0440$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PODCn	A:	FF40 4570$_H$	FF44 0450$_H$		
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$		
	B:	xxxx xxxx xxxx xx11	xxxx xxxx xx11 1111		
PDSCn	A:	FF40 4670$_H$	FF44 0460$_H$		
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$		
	B:	xxxx xxxx xxxx xx11	xxxx xxxx xx11 1111		
PISn	A:	FF40 4770$_H$	FF44 0470$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PISEn	A:	FF40 4870$_H$	FF44 0480$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xxxx xx11	xx11 1111		
PPCMDn	A:	FF40 4C70$_H$	FF44 04C0$_H$		
	I:	00$_H$	00$_H$		
	B:	1111 1111	1111 1111		
PPROTSn	A:	FF40 4B70$_H$	FF44 04B0$_H$		
	I:	00$_H$	00$_H$		
	B:	xxxx xxx1	xxxx xxx1		

2.4.5 V850E2/SK4-H Port features

This section describes the port functions and alternative functions of the V850E2/SK4-H and the registers used to control the ports.

(1) General-purpose I/O operation

The port functions and alternative functions of the V850E2/SK4-H are shown in *Table 2-38 "V850E2/SK4-H general-purpose I/O operations"*.

The mode can be selected by specifying settings for the PMCn_m, PFCn_m, PFCEn_m, and PMn_m bits.

Table 2-38 V850E2/SK4-H general-purpose I/O operations (1/4)

	Port mode		Alternative mode							
	PMCn_m = 0		PMCn_m = 1							
	Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
			PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
			ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
	Port group 0 (Always-On area, E0VDD/E0VSS power supply):									
<R>	P0_0	37	TAUJ0I0	TAUJ0O0	INTP4		$\overline{\text{CSIG0SSI}}$			IICB0SDA[b]
<R>	P0_1	38	TAUJ0I1	TAUJ0O1	INTP5			CSIG0SO[a]		IICB0SCL[b]
<R>	P0_2	39	TAUJ0I2	TAUJ0O2	INTP6		CSIG0SI			IICB1SDA[b]
<R>	P0_3	40	TAUJ0I3	TAUJ0O3	INTP7		CSIG0SC[a]			IICB1SCL[b]
<R>	P0_4	41	$\overline{\text{IEBB0RX}}$	FCN0TX	INTP8		CSIH1SI	URTE0TX		IICB2SDA[b]
<R>	P0_5	42	FCN0RX	$\overline{\text{IEBB0TX}}$	INTP9		URTE0RX	CSIH1SO[a]		IICB2SCL[b]
	P0_6	43		FCN1TX	INTP10		CSIH1RYI	CSIH1RYO	$\overline{\text{CSIH1SSI}}$	URTE10TX
	P0_7	44	FCN1RX		INTP11		CSIH1SC[a]		URTE10RX	
	P0_8	45								
	Port group 1 (Isolated area 0, E1VDD/E1VSS power supply):									
	P1_0	49	TAUA0I0	TAUA0O0	IISAACK		URTE1RX		ETH0RXER	CSIH0CSS0
	P1_1	50	TAUA0I1	TAUA0O1	IISA0SCK		$\overline{\text{CSIG4SSI}}$	URTE1TX	ETH0RXD0	CSIH0CSS1
	P1_2	51	TAUA0I2	TAUA0O2	IISA0WS			CSIG4SO[a]	ETH0RXD1	CSIH0CSS2
	P1_3	52	TAUA0I3	TAUA0O3	IISA0SDI	IISA0SDO	CSIG4SI		ETH0RXD2	CSIH0CSS3
	P1_4	53	TAUA0I4	TAUA0O4	INTP12	IISA1SDO	CSIG4SC[a]		ETH0RXD3	CSIH0CSS4
	P1_5	54	TAUA0I5	TAUA0O5	IISA1SCK		ENCA0AIN	ETH0TXD0	$\overline{\text{CSIH0SSI}}$	
	P1_6	55	TAUA0I6	TAUA0O6	IISA1WS		ENCA0BIN	ETH0TXD1	CSIH0SI	
	P1_7	58	TAUA0I7	TAUA0O7	IISA1SDI	IISA1SDO	ENCA0ZIN	ETH0TXD2		CSIH0SO[a]
	P1_8	59	TAUA0I8	TAUA0O8	INTP13	IISA2SDO	ENCA0TIN0	ETH0TXD3	CSIH0RYI	CSIH0RYO
	P1_9	60	TAUA0I9	TAUA0O9	IISA2SCK		ENCA0TIN1	ETH0TXEN	CSIH0SC[a]	
	P1_10	61	TAUA0I10	TAUA0O10	IISA2WS		ENCA1AIN	CSIH0CSS5	URTE2RX	ETH0MDC
	P1_11	62	TAUA0I11	TAUA0O11	IISA2SDI	IISA2SDO	ENCA1BIN	CSIH0CSS6	ETH0CRSDV	URTE2TX
	P1_12	63	TAUA0I12	TAUA0O12	INTP14	IISA3SDO	ENCA1ZIN	CSIH0CSS7	ETH0MDI	ETH0MDO[a]
<R>	P1_13	64	TAUA0I13	TAUA0O13	PCM0CLK[a]		ENCA1TIN0			IICB1SCL[b]
<R>	P1_14	65	TAUA0I14	TAUA0O14	PCM0SEN[a]		ENCA1TIN1	URTE3TX		IICB3SDA[b]
<R>	P1_15	66	TAUA0I15	TAUA0O15	PCM0SI	PCM0SO	URTE3RX			IICB3SCL[b]
	Port group 3 (Isolated area 0, E1VDD/E1VSS power supply):									
<R>	P3_0	67	TAUB2I1	TAUB2O1	PCM1CLK[a]		KR0I0	CSIH1CSS0		IICB0SDA[b]

Table 2-38 V850E2/SK4-H general-purpose I/O operations (2/4)

	Port mode		Alternative mode							
	PMCn_m = 0		PMCn_m = 1							
	Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
			PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
			ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
<R>	P3_1	68	TAUB2I2	TAUB2O2	PCM1SEN[a]		KR0I1	CSIH1CSS1		IICB0SCL[b]
<R>	P3_2	69	TAUB2I3	TAUB2O3	PCM1SI	PCM1SO	KR0I2	CSIH1CSS2		IICB1SDA[b]
	P3_3	72	TAUB2I5	TAUB2O5	IISAACK		KR0I3	CSIH1CSS3	ETH0REFCLK	ETH0TXER
	P3_4	73	TAUB2I6	TAUB2O6	INTP15	IISA3SDO	KR0I4	CSIH1CSS4	ETH0COL	
	P3_5	74	TAUB2I7	TAUB2O7	IISA3SCK		KR0I5	CSIH1CSS5	ETH0TXCLK	
	P3_6	75	TAUB2I9	TAUB2O9	IISA3WS		KR0I6	CSIH1CSS6	ETH0RXDV	
	P3_7	76	TAUB2I10	TAUB2O10	IISA3SDI	IISA3SDO	KR0I7	CSIH1CSS7	ETH0RXCLK	
	Port group 10 (Isolated area 0, A0VDD/A0VSS power supply):									
	P10_0	81								
	P10_1	82								
	P10_2	83								
	P10_3	84								
	P10_4	85								
	P10_5	86								
	P10_6	87								
	P10_7	88								
	P10_8	93								
	P10_9	94	ADCA0TRG0							
	P10_10	95	ADCA0TRG1							
	P10_11	96	ADCA0TRG2							
	P10_12	97								
	P10_13	98								
	P10_14	99								
	P10_15	100								
	Port group 21 (Isolated area 1, B0VDD/B0VSS power supply):									
	P21_0	129		$\overline{\text{MEMC0BEN3}}$[a]		MEMC0DQM3[a]			ETH0COL	
	P21_1	130		$\overline{\text{MEMC0BEN2}}$[a]		MEMC0DQM2[a]			ETH0CRSDV	
	P21_2	131		$\overline{\text{MEMC0BEN1}}$[a]		MEMC0DQM1[a]				
	P21_3	132		$\overline{\text{MEMC0BEN0}}$[a]		MEMC0DQM0[a]				
<R>	P21_4	133		$\overline{\text{MEMC0WR}}$[a]	$\overline{\text{IEBB0RX}}$	URTE2TX	$\overline{\text{CSIG0SSI}}$	$\overline{\text{MEMC0WE}}$[a]		IICB0SDA[b]
<R>	P21_5	134		$\overline{\text{MEMC0RD}}$[a]	URTE2RX	$\overline{\text{IEBB0TX}}$		CSIG0SO[a]		IICB0SCL[b]
<R>	P21_6	137		MEMC0CLK		URTE1TX	CSIG0SI			IICB1SDA[b]
<R>	P21_7	138	$\overline{\text{MEMC0WAIT}}$		URTE1RX		CSIG0SC[a]			IICB1SCL[b]
<R>	P21_8	139				$\overline{\text{MEMC0SDRAS}}$[a]				IICB3SDA[b]
<R>	P21_9	140		$\overline{\text{MEMC0CS2}}$		$\overline{\text{MEMC0SDCAS}}$[a]				IICB3SCL[b]
	P21_10	141		$\overline{\text{MEMC0CS3}}$	FCN1RX					
	P21_11	142		$\overline{\text{MEMC0CS4}}$		FCN1TX				
	P21_12	143		$\overline{\text{MEMC0CS7}}$			$\overline{\text{CSIG4SSI}}$			
<R>	P21_13	144	$\overline{\text{MEMC0HLDRQ}}$	$\overline{\text{MEMC0DSTB}}$[a]	FCN0RX	MEMC0CKE[a]		CSIG4SO[a]		IICB2SDA[b]

Table 2-38 V850E2/SK4-H general-purpose I/O operations (3/4)

Port mode		Alternative mode							
PMCn_m = 0		PMCn_m = 1							
Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
		PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
		ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
<R> P21_14	145		$\overline{\text{MEMC0HLDAK}}$	URTE0RX	FCN0TX	CSIG4SI			IICB2SCL[b]
P21_15	146		$\overline{\text{MEMC0ASTB}}$		URTE0TX	CSIG4SC[a]			
Port group 24 (Isolated area 1, B0VDD/B0VSS power supply):									
P24_0	154	MEMC0AD16[a]		INTP0	CSIH0CSS0	ENCA0AIN		ETH0RXER	
P24_1	155	MEMC0AD17[a]		INTP1	CSIH0CSS1	ENCA0BIN		ETH0RXD0	
P24_2	158	MEMC0AD18[a]		INTP2	CSIH0CSS2	ENCA0ZIN		ETH0RXD1	
P24_3	159	MEMC0AD19[a]		INTP3	CSIH0CSS3	ENCA0TIN0		ETH0RXD2	
P24_4	160	MEMC0AD20[a]		INTP4	CSIH0CSS4	ENCA0TIN1		ETH0RXD3	
P24_5	161	MEMC0AD21[a]		INTP5	CSIH0CSS5				ETH0TXD0
P24_6	162	MEMC0AD22[a]		INTP6	CSIH0CSS6				ETH0TXD1
P24_7	165	MEMC0AD23[a]		INTP7	CSIH0CSS7				ETH0TXD2
P24_8	166	MEMC0AD24[a]		INTP8	CSIH1CSS0	ENCA1AIN			ETH0TXD3
P24_9	167	MEMC0AD25[a]		INTP9	CSIH1CSS1	ENCA1BIN			ETH0TXEN
P24_10	168	MEMC0AD26[a]		INTP10	CSIH1CSS2	ENCA1ZIN			ETH0MDC
P24_11	169	MEMC0AD27[a]		INTP11	CSIH1CSS3	ENCA1TIN0		ETH0REFCLK	ETH0TXER
P24_12	170	MEMC0D28[a]		INTP12	CSIH1CSS4	ENCA1TIN1		ETH0MDI	ETH0MDO[a]
P24_13	171	MEMC0D29[a]		INTP13	CSIH1CSS5			ETH0TXCLK	
P24_14	172	MEMC0D30[a]		INTP14	CSIH1CSS6			ETH0RXDV	
P24_15	173	MEMC0D31[a]		INTP15	CSIH1CSS7			ETH0RXCLK	
Port group 25 (Isolated area 1, B0VDD/B0VSS power supply):									
P25_0	174	MEMC0AD0[a]		TAUA0I0	TAUA0O0	IISAACK	IISA0SDO	$\overline{\text{CSIH0SSI}}$	
P25_1	175	MEMC0AD1[a]		TAUA0I1	TAUA0O1	IISA0SCK		CSIH0SI	
P25_2	176	MEMC0AD2[a]		TAUA0I2	TAUA0O2	IISA0WS			CSIH0SO[a]
P25_3	1	MEMC0AD3[a]		TAUA0I3	TAUA0O3	IISA0SDI	IISA0SDO	CSIH0RYI	CSIH0RYO
P25_4	2	MEMC0AD4[a]		TAUA0I4	TAUA0O4		IISA1SDO	CSIH0SC[a]	
P25_5	3	MEMC0AD5[a]		TAUA0I5	TAUA0O5	IISA1SCK			
P25_6	4	MEMC0AD6[a]		TAUA0I6	TAUA0O6	IISA1WS			
P25_7	5	MEMC0AD7[a]		TAUA0I7	TAUA0O7	IISA1SDI	IISA1SDO		
P25_8	8	MEMC0AD8[a]		TAUA0I8	TAUA0O8		IISA2SDO	$\overline{\text{CSIH1SSI}}$	
P25_9	9	MEMC0AD9[a]		TAUA0I9	TAUA0O9	IISA2SCK		CSIH1SI	
P25_10	10	MEMC0AD10[a]		TAUA0I10	TAUA0O10	IISA2WS			CSIH1SO[a]
P25_11	11	MEMC0AD11[a]		TAUA0I11	TAUA0O11	IISA2SDI	IISA2SDO	CSIH1RYI	CSIH1RYO
P25_12	12	MEMC0AD12[a]		TAUA0I12	TAUA0O12		IISA3SDO	CSIH1SC[a]	
P25_13	13	MEMC0AD13[a]		TAUA0I13	TAUA0O13	IISA3SCK			
P25_14	14	MEMC0AD14[a]		TAUA0I14	TAUA0O14	IISA3WS			
P25_15	15	MEMC0AD15[a]		TAUA0I15	TAUA0O15	IISA3SDI	IISA3SDO		
Port group 26 (Isolated area 1, B0VDD/B0VSS power supply):									
P26_0	103	KR0I0	MEMC0A0[a]	TAUB2I0	TAUB2O0		IISA4SDO	$\overline{\text{CSIH2SSI}}$	
P26_1	104	KR0I1	MEMC0A1[a]	TAUB2I1	TAUB2O1	IISA4SCK		CSIH2SI	

Table 2-38 V850E2/SK4-H general-purpose I/O operations (4/4)

Port mode		Alternative mode							
PMCn_m = 0		PMCn_m = 1							
Pin name	Pin No.	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
		PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
		ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
P26_2	105	KR0I2	MEMC0A2[a]	TAUB2I2	TAUB2O2	IISA4WS			CSIH2SO[a]
P26_3	106	KR0I3	MEMC0A3[a]	TAUB2I3	TAUB2O3	IISA4SDI	IISA4SDO	CSIH2RYI	CSIH2RYO
P26_4	107	KR0I4	MEMC0A4[a]	TAUB2I4	TAUB2O4		IISA5SDO	CSIH2SC[a]	
P26_5	108	KR0I5	MEMC0A5[a]	TAUB2I5	TAUB2O5	IISA5SCK			CSIH2CSS0
P26_6	109	KR0I6	MEMC0A6[a]	TAUB2I6	TAUB2O6	IISA5WS			CSIH2CSS1
P26_7	110	KR0I7	MEMC0A7[a]	TAUB2I7	TAUB2O7	IISA5SDI	IISA5SDO		CSIH2CSS2
P26_8	111	INTP8	MEMC0A8[a]	TAUB2I8	TAUB2O8	IISA3SCK			CSIH2CSS3
P26_9	114	INTP9	MEMC0A9[a]	TAUB2I9	TAUB2O9	IISA3WS			CSIH2CSS4
P26_10	115	INTP10	MEMC0A10[a]	TAUB2I10	TAUB2O10	IISA2SCK			CSIH2CSS5
P26_11	116	INTP11	MEMC0A11[a]	TAUB2I11	TAUB2O11	IISA2WS			CSIH2CSS6
P26_12	117	INTP12	MEMC0A12[a]	TAUB2I12	TAUB2O12	IISA0SCK			CSIH2CSS7
P26_13	118	INTP13	MEMC0A13[a]	TAUB2I13	TAUB2O13	IISA0WS			
P26_14	119	INTP14	MEMC0A14[a]	TAUB2I14	TAUB2O14	IISA5SDI	IISA5SDO		
P26_15	120	INTP15	MEMC0A15[a]	TAUB2I15	TAUB2O15	IISA4SDI	IISA4SDO		
Port group 27 (Isolated area 1, B0VDD/B0VSS power supply):									
P27_0	121	INTP0	MEMC0A16[a]			IISA3SDI	IISA3SDO		CSIH1CSS0
P27_1	122	INTP1	MEMC0A17[a]			IISA2SDI	IISA2SDO		CSIH1CSS1
P27_2	123	INTP2	MEMC0A18[a]			IISA1SDI	IISA1SDO		CSIH1CSS2
P27_3	126	INTP3	MEMC0A19[a]			IISA0SDI	IISA0SDO		CSIH1CSS3
P27_4	127	INTP4	MEMC0A20[a]			IISA0SCK			URTE10TX
P27_5	128	INTP5	MEMC0A21[a]			IISA0WS		URTE10RX	
Port group 28 (Isolated area 1, B0VDD/B0VSS power supply):									
P28_0	152	INTP6	MEMC0A22[a]		MEMC0A24[a]	IISA1SCK			
P28_1	153	INTP7	MEMC0A23[a]		MEMC0A25[a]	IISA1WS			
Port group JP0 (Always-On area, E0VDD/E0VSS power supply):[c]									
JP0_0	22	INTP0		TAUJ0I0	TAUJ0O0				
JP0_1	23	INTP1		TAUJ0I1	TAUJ0O1				
JP0_2	24	INTP2		TAUJ0I2	TAUJ0O2				
JP0_3	25	INTP3		TAUJ0I3	TAUJ0O3				
JP0_4	20								
JP0_5	26	NMI	RTCA0OUT						

a) The PIPCn.PIPCn_m bit must be set to 1 to use this alternative mode. The mode can then be set to I/O, after which the PMn.PMn_m bit setting is ignored.

<R> b) When using the following alternate-function pins, set the PBDCn.PBDCnm bit to 1.

c) JP0_0 to JP0_5 are used for debugging. For details, see *2.4.2 "Overview of port features" on page 76*.

(2) V850E2/SK4-H port control registers

This section describes the registers used to control the ports in the V850E2/SK4-H and their addresses and initial values.

Key for symbols in tables

A: Register address

I: Initial value

B: Valid bit

- 1: Valid, X: Invalid
- Right side: Bit 0, left side: Bit 15

Table 2-39 V850E2/SK4-H port control registers (groups 0, 1, 3, 10) (1/2)

Register		Port group n =			
		0	1	3	10
Pn	A:	FF40 0000$_H$	FF40 0004$_H$	FF40 000C$_H$	FF40 0028$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PSRn	A:	FF40 0100$_H$	FF40 0104$_H$	FF40 010C$_H$	FF40 0128$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PNOTn	A:	FF40 0700$_H$	FF40 0704$_H$	FF40 070C$_H$	FF40 0728$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PPRn	A:	FF40 0200$_H$	FF40 0204$_H$	FF40 020C$_H$	FF40 0228$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PMn	A:	FF40 0300$_H$	FF40 0304$_H$	FF40 030C$_H$	FF40 0328$_H$
	I:	FFFE$_H$	FFFF$_H$	FFFF$_H$	FFFF$_H$
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PMCn	A:	FF40 0400$_H$	FF40 0404$_H$	FF40 040C$_H$	FF40 0428$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxxx 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	xxxx 111x xxxx xxxx
PFCn	A:	FF40 0500$_H$	FF40 0504$_H$	FF40 050C$_H$	–
	I:	0000$_H$	0000$_H$	0000$_H$	
	B:	xxxx xxxx 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	
PFCEn	A:	FF40 0600$_H$	FF40 0604$_H$	FF40 060C$_H$	–
	I:	0000$_H$	0000$_H$	0000$_H$	
	B:	xxxx xxxx 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	
PMSRn	A:	FF40 0800$_H$	FF40 0804$_H$	FF40 080C$_H$	FF40 0828$_H$
	I:	0000 FFFE$_H$	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PMCSRn	A:	FF40 0900$_H$	FF40 0904$_H$	FF40 090C$_H$	FF40 0928$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	xxxx xxxx 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	xxxx 111x xxxx xxxx

Table 2-39 V850E2/SK4-H port control registers (groups 0, 1, 3, 10) (2/2)

Register		Port group n =			
		0	1	3	10
PIBCn	A:	FF40 4000$_H$	FF40 4004$_H$	FF40 400C$_H$	FF40 4028$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PBDCn	A:	FF40 4100$_H$	FF40 4104$_H$	FF40 410C$_H$	FF40 4128$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PIPCn	A:	FF40 4200$_H$	FF40 4204$_H$	FF40 420C$_H$	–
	I:	0000$_H$	0000$_H$	0000$_H$	
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	
PUn	A:	FF40 4300$_H$	FF40 4304$_H$	FF40 430C$_H$	–
	I:	0000$_H$	0000$_H$	0000$_H$	
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	
PDn	A:	FF40 4400$_H$	FF40 4404$_H$	FF40 440C$_H$	–
	I:	0000$_H$	0000$_H$	0000$_H$	
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	
PODCn	A:	FF40 4500$_H$	FF40 4504$_H$	FF40 450C$_H$	FF40 4528$_H$
	I:	0000 0000 0000 0001$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PDSCn	A:	FF40 4600$_H$	FF40 4604$_H$	FF40 460C$_H$	–
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	
PISn	A:	FF40 4700$_H$	FF40 4704$_H$	FF40 470C$_H$	–
	I:	0000$_H$	0000$_H$	0000$_H$	
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	
PISEn	A:	FF40 4800$_H$	FF40 4804$_H$	FF40 480C$_H$	–
	I:	0000$_H$	0000$_H$	0000$_H$	
	B:	xxxx xxx1 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	
PPCMDn	A:	FF40 4C00$_H$	FF40 4C04$_H$	FF40 4C0C$_H$	FF40 4C28$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	1111 1111	1111 1111	1111 1111	1111 1111
PPROTSn	A:	FF40 4B00$_H$	FF40 4B04$_H$	FF40 4B0C$_H$	FF40 4B28$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-40 V850E2/SK4-H port control registers (groups 21, 24 to 26) (1/2)

Register		Port group n =			
		21	24	25	26
Pn	A:	FF40 0054$_H$	FF40 0060$_H$	FF40 0064$_H$	FF40 0068$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PSRn	A:	FF40 0154$_H$	FF40 0160$_H$	FF40 0164$_H$	FF40 0168$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PNOTn	A:	FF40 0754$_H$	FF40 0760$_H$	FF40 0764$_H$	FF40 0768$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PPRn	A:	FF40 0254$_H$	FF40 0260$_H$	FF40 0264$_H$	FF40 0268$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PMn	A:	FF40 0354$_H$	FF40 0360$_H$	FF40 0364$_H$	FF40 0368$_H$
	I:	FFFF$_H$	FFFF$_H$	FFFF$_H$	FFFF$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PMCn	A:	FF40 0454$_H$	FF40 0460$_H$	FF40 0464$_H$	FF40 0468$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PFCn	A:	FF40 0554$_H$	FF40 0560$_H$	FF40 0564$_H$	FF40 0568$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>	B:	111x 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PFCEn	A:	FF40 0654$_H$	FF40 0660$_H$	FF40 0664$_H$	FF40 0668$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
<R>	B:	1111 xx11 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PMSRn	A:	FF40 0854$_H$	FF40 0860$_H$	FF40 0864$_H$	FF40 0868$_H$
	I:	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PMCSRn	A:	FF40 0954$_H$	FF40 0960$_H$	FF40 0964$_H$	FF40 0968$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PIBCn	A:	FF40 4054$_H$	FF40 4060$_H$	FF40 4064$_H$	FF40 4068$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PBDCn	A:	FF40 4154$_H$	FF40 4160$_H$	FF40 4164$_H$	FF40 4168$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PIPCn	A:	FF40 4254$_H$	FF40 4260$_H$	FF40 4264$_H$	FF40 4268$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111

Table 2-40 V850E2/SK4-H port control registers (groups 21, 24 to 26) (2/2)

Register		Port group n =			
		21	24	25	26
PUn	A:	FF40 4354$_H$	FF40 4360$_H$	FF40 4364$_H$	FF40 4368$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PDn	A:	FF40 4454$_H$	FF40 4460$_H$	FF40 4464$_H$	FF40 4468$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PODCn	A:	FF40 4554$_H$	FF40 4560$_H$	FF40 4564$_H$	FF40 4568$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PDSCn	A:	FF40 4654$_H$	FF40 4660$_H$	FF40 4664$_H$	FF40 4668$_H$
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PISn	A:	FF40 4754$_H$	FF40 4760$_H$	FF40 4764$_H$	FF40 4768$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PISEn	A:	FF40 4854$_H$	FF40 4860$_H$	FF40 4864$_H$	FF40 4868$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PPCMDn	A:	FF40 4C54$_H$	FF40 4C60$_H$	FF40 4C64$_H$	FF40 4C68$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	1111 1111	1111 1111	1111 1111	1111 1111
PPROTSn	A:	FF40 4B54$_H$	FF40 4B60$_H$	FF40 4B64$_H$	FF40 4B68$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-41 V850E2/SK4-H port control registers (groups 27, 28, JP0) (1/2)

Register		Port group n =			
		27	28	JP0	
Pn	A:	FF40 006C$_H$	FF40 0070$_H$	FF44 0000$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PSRn	A:	FF40 016C$_H$	FF40 0170$_H$	FF44 0010$_H$	
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xxxx xxxx xx11 1111	
PNOTn	A:	FF40 076C$_H$	FF40 0770$_H$	FF44 0070$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PPRn	A:	FF40 026C$_H$	FF40 0270$_H$	FF44 0020$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PMn	A:	FF40 036C$_H$	FF40 0370$_H$	FF44 0030$_H$	
	I:	FFFF$_H$	FFFF$_H$	FF$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PMCn	A:	FF40 046C$_H$	FF40 0470$_H$	FF44 0040$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx1x 1111	
PFCn	A:	FF40 056C$_H$	FF40 0570$_H$	FF44 0050$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xxxx 1111	
PFCEn	A:	FF40 066C$_H$	FF40 0670$_H$	–	
	I:	0000$_H$	0000$_H$		
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11		
PMSRn	A:	FF40 086C$_H$	FF40 0870$_H$	FF44 0080$_H$	
	I:	0000 0000 0000 FFFF$_H$	0000 0000 0000 FFFF$_H$	0000 0000 0000 00FF$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xxxx xxxx xx11 1111	
PMCSRn	A:	FF40 096C$_H$	FF40 0970$_H$	FF44 0090$_H$	
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xxxx xxxx xx1x 1111	
PIBCn	A:	FF40 406C$_H$	FF40 4070$_H$	FF44 0400$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PBDCn	A:	FF40 416C$_H$	FF40 4170$_H$	FF44 0410$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PIPCn	A:	FF40 426C$_H$	FF40 4270$_H$	FF44 0420$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	

Table 2-41 V850E2/SK4-H port control registers (groups 27, 28, JP0) (2/2)

Register		Port group n =			
		27	28	JP0	
PUn	A:	FF40 436C$_H$	FF40 4370$_H$	FF44 0430$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PDn	A:	FF40 446C$_H$	FF40 4470$_H$	FF44 0440$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PODCn	A:	FF40 456C$_H$	FF40 4570$_H$	FF44 0450$_H$	
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xxxx xxxx xx11 1111	
PDSCn	A:	FF40 466C$_H$	FF40 4670$_H$	FF44 0460$_H$	
	I:	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	0000 0000 0000 0000$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xxxx xxxx xx11 1111	
PISn	A:	FF40 476C$_H$	FF40 4770$_H$	FF44 0470$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PISEn	A:	FF40 486C$_H$	FF40 4870$_H$	FF44 0480$_H$	
	I:	0000$_H$	0000$_H$	00$_H$	
	B:	xxxx xxxx xx11 1111	xxxx xxxx xxxx xx11	xx11 1111	
PPCMDn	A:	FF40 4C6C$_H$	FF40 4C70$_H$	FF44 04C0$_H$	
	I:	00$_H$	00$_H$	00$_H$	
	B:	1111 1111	1111 1111	1111 1111	
PPROTSn	A:	FF40 4B6C$_H$	FF40 4B70$_H$	FF44 04B0$_H$	
	I:	00$_H$	00$_H$	00$_H$	
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	

2.4.6 List of pin functions in alphabetical order

The names and operations of the alternative function pins are shown in alphabetical order in the tables below.

Note These tables show all the signals used in the V850E2/Sx4-H, but whether a specific alternative function is available in a specific device is not noted.

Table 2-42 List of pin functions in alphabetical order (1/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
A0VDD	TBD	TBD	89	–	Positive power supply for A/D converter	–
A0VSS	TBD	TBD	90	–	Ground potential for A/D converter	–
ADCA0I0	TBD	TBD	81	Input	Input of analog signals to A/D converter	P10_0
ADCA0I1	TBD	TBD	82			P10_1
ADCA0I2	TBD	TBD	83			P10_2
ADCA0I3	TBD	TBD	84			P10_3
ADCA0I4	TBD	TBD	85			P10_4
ADCA0I5	TBD	TBD	86			P10_5
ADCA0I6	TBD	TBD	87			P10_6
ADCA0I7	TBD	TBD	88			P10_7
ADCA0I8	TBD	TBD	93			P10_8
ADCA0I9	TBD	TBD	94			P10_9/ADCA0TRG0
ADCA0I10	TBD	TBD	95			P10_10/ADCA0TRG1
ADCA0I11	TBD	TBD	96			P10_11/ADCA0TRG2
ADCA0I12	TBD	TBD	97			P10_12
ADCA0I13	TBD	TBD	98			P10_13
ADCA0I14	TBD	TBD	99			P10_14
ADCA0I15	TBD	TBD	100			P10_15
ADCA0TRG0	TBD	TBD	94	Input	A/D converter external trigger input	P10_9/ADCA0I9
ADCA0TRG1	TBD	TBD	95			P10_10/ADCA0I10
ADCA0TRG2	TBD	TBD	96			P10_11/ADCA0I11
<R> A0VREFM	TBD	TBD	92	–	Reference voltage for A/D converter	–
<R> A0VREFP	TBD	TBD	91	–		–
B0VDD	TBD	TBD	7	–	Positive power supply for bus interface and alternate-function ports	–
	TBD	TBD	113			–
	TBD	TBD	136			–
	TBD	TBD	164			–
B0VSS	TBD	TBD	6	–	Ground potential for bus interface and alternate-function ports	–
	TBD	TBD	112			–
	TBD	TBD	135			–
	TBD	TBD	163			–
CSIG0SC	TBD	TBD	40	I/O	CSIG0 serial clock	P0_3/TAUJ0I3/TAUJ0O3/INTP7/IICB1SCL
	TBD	TBD	138			P21_7/$\overline{\text{MEMC0WAIT}}$/URTE1RX/IICB1SCL
CSIG0SI	TBD	TBD	39	Input	CSIG0 serial data input	P0_2/TAUJ0I2/TAUJ0O2/INTP6/IICB1SDA
	TBD	TBD	137			P21_6/MEMC0CLK/URTE1TX/IICB1SDA

Table 2-42 List of pin functions in alphabetical order (2/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
CSIG0SO	TBD	TBD	38	Output	CSIG0 serial data output	P0_1/TAUJ0I1/TAUJ0O1/INTP5/ IICB0SCL/FLMD1
	TBD	TBD	134			P21_5/$\overline{\text{MEMC0RD}}$/URTE2RX/ $\overline{\text{IEBB0TX}}$/IICB0SCL
$\overline{\text{CSIG0SSI}}$	TBD	TBD	37	Input	CSIG0 slave selection input	P0_0/TAUJ0I0/TAUJ0O0/INTP4/ IICB0SDA/$\overline{\text{RESETOUT}}$
	TBD	TBD	133			P21_4/$\overline{\text{MEMC0WR}}$/$\overline{\text{IEBB0RX}}$/ URTE2TX/$\overline{\text{MEMC0WE}}$/IICB0SDA
CSIG4SC	TBD	TBD	53	I/O	CSIG4 serial clock	P1_4/TAUA0I4/TAUA0O4/INTP12/ IISA1SDO/ETH0RXD3/CSIH0CSS4
	TBD	TBD	146			P21_15/$\overline{\text{MEMC0ASTB}}$/URTE0TX
CSIG4SI	TBD	TBD	52	Input	CSIG4 serial data input	P1_3/TAUA0I3/TAUA0O3/IISA0SDI/ IISA0SDO/ETH0RXD2/CSIH0CSS3
	TBD	TBD	145			P21_14/$\overline{\text{MEMC0HLDAK}}$/URTE0RX/ FCN0TX/IICB2SCL
CSIG4SO	TBD	TBD	51	Output	CSIG4 serial data output	P1_2/TAUA0I2/TAUA0O2/IISA0WS/ ETH0RXD1/CSIH0CSS2
	TBD	TBD	144			P21_13/$\overline{\text{MEMC0HLDRQ}}$/ $\overline{\text{MEMC0DSTB}}$/FCN0RX/MEMC0CKE/ IICB2SDA
$\overline{\text{CSIG4SSI}}$	TBD	TBD	50	Input	CSIG4 slave selection input	P1_1/TAUA0I1/TAUA0O1/IISA0SCK/ URTE1TX/ETH0RXD0/CSIH0CSS1
	TBD	TBD	143			P21_12/$\overline{\text{MEMC0CS7}}$
CSIH0CSS0	TBD	TBD	49	Output	CSIH0 chip select output	P1_0/TAUA0I0/TAUA0O0/IISAACK/ URTE1RX/ETH0RXER
	TBD	TBD	154			P24_0/MEMC0AD16/INTP0/ ENCA0AIN/ETH0RXER
CSIH0CSS1	TBD	TBD	50			P1_1/TAUA0I1/TAUA0O1/IISA0SCK/ $\overline{\text{CSIG4SSI}}$/URTE1TX/ETH0RXD0
	TBD	TBD	155			P24_1/MEMC0AD17/INTP1/ ENCA0BIN/ETH0RXD0
CSIH0CSS2	TBD	TBD	51			P1_2/TAUA0I2/TAUA0O2/IISA0WS/ CSIG4SO/ETH0RXD1
	TBD	TBD	158			P24_2/MEMC0AD18/INTP2/ ENCA0ZIN/ETH0RXD1
CSIH0CSS3	TBD	TBD	52			P1_3/TAUA0I3/TAUA0O3/IISA0SDI/ IISA0SDO/CSIG4SI/ETH0RXD2
	TBD	TBD	159			P24_3/MEMC0AD19/INTP3/ ENCA0TIN0/ETH0RXD2
CSIH0CSS4	TBD	TBD	53			P1_4/TAUA0I4/TAUA0O4/INTP12/ IISA1SDO/CSIG4SC/ETH0RXD3
	TBD	TBD	160			P24_4/MEMC0AD20/INTP4/ ENCA0TIN1/ETH0RXD3
CSIH0CSS5	TBD	TBD	61			P1_10/TAUA0I10/TAUA0O10/IISA2WS/ ENCA1AIN/URTE2RX/ETH0MDC
	TBD	TBD	161			P24_5/MEMC0AD21/INTP5/ ETH0TXD0

Table 2-42 List of pin functions in alphabetical order (3/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
CSIH0CSS6	TBD	TBD	62	Output	CSIH0 chip select output	P1_11/TAUA0I11/TAUA0O11/ IISA2SDI/IISA2SDO/ENCA1BIN/ ETH0CRSDV/URTE2TX
	TBD	TBD	162			P24_6/MEMC0AD22/INTP6/ ETH0TXD1
CSIH0CSS7	TBD	TBD	63			P1_12/TAUA0I12/TAUA0O12/INTP14/ IISA3SDO/ENCA1ZIN/ETH0MDI/ ETH0MDO
	TBD	TBD	165			P24_7/MEMC0AD23/INTP7/ ETH0TXD2
CSIH0RYI	TBD	TBD	1	Input	CSIH0 ready/busy signal	P25_3/MEMC0AD3/TAUA0I3/ TAUA0O3/IISA0SDI/IISA0SDO/ CSIH0RYO
	TBD	TBD	59			P1_8/TAUA0I8/TAUA0O8/INTP13/ IISA2SDO/ENCA0TIN0/ETH0TXD3/ CSIH0RYO
CSIH0RYO	TBD	TBD	1	Output	CSIH0 ready/busy signal	P25_3/MEMC0AD3/TAUA0I3/ TAUA0O3/IISA0SDI/IISA0SDO/ CSIH0RYI
	TBD	TBD	59			P1_8/TAUA0I8/TAUA0O8/INTP13/ IISA2SDO/ENCA0TIN0/ETH0TXD3/ CSIH0RYI
CSIH0SC	TBD	TBD	2	I/O	CSIH0 serial clock	P25_4/MEMC0AD4/TAUA0I4/ TAUA0O4/IISA1SDO
	TBD	TBD	60			P1_9/TAUA0I9/TAUA0O9/IISA2SCK/ ENCA0TIN1/ETH0TXEN
CSIH0SI	TBD	TBD	55	Input	CSIH0 serial data input	P1_6/TAUA0I6/TAUA0O6/IISA1WS/ ENCA0BIN/ETH0TXD1
	TBD	TBD	175			P25_1/MEMC0AD1/TAUA0I1/ TAUA0O1/IISA0SCK
CSIH0SO	TBD	TBD	58	Output	CSIH0 serial data output	P1_7/TAUA0I7/TAUA0O7/IISA1SDI/ IISA1SDO/ENCA0ZIN/ETH0TXD2
	TBD	TBD	176			P25_2/MEMC0AD2/TAUA0I2/ TAUA0O2/IISA0WS
$\overline{\text{CSIH0SSI}}$	TBD	TBD	54	Input	CSIH0 slave selection input	P1_5/TAUA0I5/TAUA0O5/IISA1SCK/ ENCA0AIN/ETH0TXD0
	TBD	TBD	174			P25_0/MEMC0AD0/TAUA0I0/ TAUA0O0/IISAACK/IISA0SDO
CSIH1CSS0	TBD	TBD	67	Output	CSIH1 chip select output	P3_0/TAUB2I1/TAUB2O1/PCM1CLK/ KR0I0/IICB0SDA
	TBD	TBD	121			P27_0/INTP0/MEMC0A16/IISA3SDI/ IISA3SDO
	TBD	TBD	166			P24_8/MEMC0AD24/INTP8/ ENCA1AIN/ETH0TXD3

Table 2-42 List of pin functions in alphabetical order (4/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
CSIH1CSS1	TBD	TBD	68	Output	CSIH1 chip select output	P3_1/TAUB2I2/TAUB2O2/PCM1SEN/ KR0I1/IICB0SCL
	TBD	TBD	122			P27_1/INTP1/MEMC0A17/IISA2SDI/ IISA2SDO
	TBD	TBD	167			P24_9/MEMC0AD25/INTP9/ ENCA1BIN/ETH0TXEN
CSIH1CSS2	TBD	TBD	69			P3_2/TAUB2I3/TAUB2O3/PCM1SI/ PCM1SO/KR0I2/IICB1SDA
	TBD	TBD	123			P27_2/INTP2/MEMC0A18/IISA1SDI/ IISA1SDO
	TBD	TBD	168			P24_10/MEMC0AD26/INTP10/ ENCA1ZIN/ETH0MDC
CSIH1CSS3	TBD	TBD	72			P3_3/TAUB2I5/TAUB2O5/IISAACK/ KR0I3/ETH0REFCLK/ETH0TXER
	TBD	TBD	126			P27_3/INTP3/MEMC0A19/IISA0SDI/ IISA0SDO
	TBD	TBD	169			P24_11/MEMC0AD27/INTP11/ ENCA1TIN0/ETH0REFCLK/ ETH0TXER
CSIH1CSS4	TBD	TBD	73			P3_4/TAUB2I6/TAUB2O6/INTP15/ IISA3SDO/KR0I4/ETH0COL
	TBD	TBD	170			P24_12/MEMC0D28/INTP12/ ENCA1TIN1/ETH0MDI/ETH0MDO
CSIH1CSS5	TBD	TBD	74			P3_5/TAUB2I7/TAUB2O7/IISA3SCK/ KR0I5/ETH0TXCLK
	TBD	TBD	171			P24_13/MEMC0D29/INTP13/ ETH0TXCLK
CSIH1CSS6	TBD	TBD	75			P3_6/TAUB2I9/TAUB2O9/IISA3WS/ KR0I6/ETH0RXDV
	TBD	TBD	172			P24_14/MEMC0D30/INTP14/ ETH0RXDV
CSIH1CSS7	TBD	TBD	76			P3_7/TAUB2I10/TAUB2O10/IISA3SDI/ IISA3SDO/KR0I7/ETH0RXCLK
	TBD	TBD	173			P24_15/MEMC0D31/INTP15/ ETH0RXCLK
CSIH1RYI	TBD	TBD	11	Input	CSIH1 ready/busy signal	P25_11/MEMC0AD11/TAUA0I11/ TAUA0O11/IISA2SDI/IISA2SDO/ CSIH1RYO
	TBD	TBD	43			P0_6/FCN1TX/INTP10/$\overline{\text{CSIH1SSI}}$/ URTE10TX/CSIH1RYO
CSIH1RYO	TBD	TBD	11	Output	CSIH1 ready/busy signal	P25_11/MEMC0AD11/TAUA0I11/ TAUA0O11/IISA2SDI/IISA2SDO/ CSIH1RYI
	TBD	TBD	43			P0_6/FCN1TX/INTP10/$\overline{\text{CSIH1SSI}}$/ URTE10TX/CSIH1RYI
CSIH1SC	TBD	TBD	12	I/O	CSIH1 serial clock	P25_12/MEMC0AD12/TAUA0I12/ TAUA0O12/IISA3SDO
	TBD	TBD	44			P0_7/FCN1RX/INTP11/URTE10RX

Table 2-42 List of pin functions in alphabetical order (5/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
CSIH1SI	TBD	TBD	9	Input	CSIH1 serial data input	P25_9/MEMC0AD9/TAUA0I9/ TAUA0O9/IISA2SCK
	TBD	TBD	41			P0_4/$\overline{\text{IEBB0RX}}$/FCN0TX/INTP8/ URTE0TX/IICB2SDA
CSIH1SO	TBD	TBD	10	Output	CSIH1 serial data output	P25_10/MEMC0AD10/TAUA0I10/ TAUA0O10/IISA2WS
	TBD	TBD	42			P0_5/FCN0RX/$\overline{\text{IEBB0TX}}$/INTP9/ URTE0RX/IICB2SCL
$\overline{\text{CSIH1SSI}}$	TBD	TBD	8	Input	CSIH1 slave selection input	P25_8/MEMC0AD8/TAUA0I8/ TAUA0O8/IISA2SDO
	TBD	TBD	43			P0_6/FCN1TX/INTP10/CSIH1RYI/ CSIH1RYO/URTE10TX
CSIH2CSS0	TBD	TBD	108	Output	CSIH2 chip select output	P26_5/KR0I5/MEMC0A5/TAUB2I5/ TAUB2O5/IISA5SCK
CSIH2CSS1	TBD	TBD	109			P26_6/KR0I6/MEMC0A6/TAUB2I6/ TAUB2O6/IISA5WS
CSIH2CSS2	TBD	TBD	110			P26_7/KR0I7/MEMC0A7/TAUB2I7/ TAUB2O7/IISA5SDI/IISA5SDO
CSIH2CSS3	TBD	TBD	111			P26_8/INTP8/MEMC0A8/TAUB2I8/ TAUB2O8/IISA3SCK
CSIH2CSS4	TBD	TBD	114			P26_9/INTP9/MEMC0A9/TAUB2I9/ TAUB2O9/IISA3WS
CSIH2CSS5	TBD	TBD	115			P26_10/INTP10/MEMC0A10/ TAUB2I10/TAUB2O10/IISA2SCK
CSIH2CSS6	TBD	TBD	116			P26_11/INTP11/MEMC0A11/ TAUB2I11/TAUB2O11/IISA2WS
CSIH2CSS7	TBD	TBD	117			P26_12/INTP12/MEMC0A12/ TAUB2I12/TAUB2O12/IISA0SCK
CSIH2RYI	TBD	TBD	106	Input	CSIH2 ready/busy signal	P26_3/KR0I3/MEMC0A3/TAUB2I3/ TAUB2O3/IISA4SDI/IISA4SDO/ CSIH2RYO
CSIH2RYO	TBD	TBD	106	Output	CSIH2 ready/busy signal	P26_3/KR0I3/MEMC0A3/TAUB2I3/ TAUB2O3/IISA4SDI/IISA4SDO/ CSIH2RYI
CSIH2SC	TBD	TBD	107	I/O	CSIH2 serial clock	P26_4/KR0I4/MEMC0A4/TAUB2I4/ TAUB2O4/IISA5SDO
CSIH2SI	TBD	TBD	104	Input	CSIH2 serial data input	P26_1/KR0I1/MEMC0A1/TAUB2I1/ TAUB2O1/IISA4SCK
CSIH2SO	TBD	TBD	105	Output	CSIH2 serial data output	P26_2/KR0I2/MEMC0A2/TAUB2I2/ TAUB2O2/IISA4WS
$\overline{\text{CSIH2SSI}}$	TBD	TBD	103	Input	CSIH2 slave selection input	P26_0/KR0I0/MEMC0A0/TAUB2I0/ TAUB2O0/IISA4SDO
CVDD	TBD	TBD	47	–	Positive power supply for internal circuits	–
	TBD	TBD	78			–
	TBD	TBD	101			–
	TBD	TBD	124			–
	TBD	TBD	156			–

Table 2-42 List of pin functions in alphabetical order (6/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
CVSS	TBD	TBD	48	–	Ground potential for internal circuits	–
	TBD	TBD	102			–
	TBD	TBD	125			–
	TBD	TBD	157			–
$\overline{\text{DCURDY}}$	TBD	TBD	26	Input	Debug ready signal	JP0_5/NMI/RTCA0OUT
DCUTCK	TBD	TBD	24	Input	Debug clock input	JP0_2/INTP2/TAUJ0I2/TAUJ0O2
DCUTDI	TBD	TBD	22	Input	Debug data input	JP0_0/INTP0/TAUJ0I0/TAUJ0O0
DCUTDO	TBD	TBD	23	Output	Debug data output	JP0_1/INTP1/TAUJ0I1/TAUJ0O1
DCUTMS	TBD	TBD	25	Input	Debug mode selection	JP0_3/INTP3/TAUJ0I3/TAUJ0O3
$\overline{\text{DCUTRST}}$	TBD	TBD	20	Input	Debug reset	JP0_4
E0VDD	TBD	TBD	34	–	Positive power supply for ports	–
E1VDD	TBD	TBD	56			–
	TBD	TBD	70			–
E0VSS	TBD	TBD	35	–	Ground potential for ports	–
E1VSS	TBD	TBD	57			–
	TBD	TBD	71			–
ENCA0AIN	TBD	TBD	54	Input	ENCA0 encoder input (phase A)	P1_5/TAUA0I5/TAUA0O5/IISA1SCK/ ETH0TXD0/$\overline{\text{CSIH0SSI}}$
	TBD	TBD	154			P24_0/MEMC0AD16/INTP0/ CSIH0CSS0/ETH0RXER
ENCA0BIN	TBD	TBD	55	Input	ENCA0 encoder input (phase B)	P1_6/TAUA0I6/TAUA0O6/IISA1WS/ ETH0TXD1/CSIH0SI
	TBD	TBD	155			P24_1/MEMC0AD17/INTP1/ CSIH0CSS1/ETH0RXD0
ENCA0TIN0	TBD	TBD	59	Input	ENCA0 capture trigger input	P1_8/TAUA0I8/TAUA0O8/INTP13/ IISA2SDO/ETH0TXD3/CSIH0RYO
	TBD	TBD	159			P24_3/MEMC0AD19/INTP3/ CSIH0CSS3/ETH0RXD2
ENCA0TIN1	TBD	TBD	60			P1_9/TAUA0I9/TAUA0O9/IISA2SCK/ ETH0TXEN/CSIH0SC
	TBD	TBD	160			P24_4/MEMC0AD20/INTP4/ CSIH0CSS4/ETH0RXD3
ENCA0ZIN	TBD	TBD	58	Input	ENCA0 encoder input (phase Z)	P1_7/TAUA0I7/TAUA0O7/IISA1SDI/ IISA1SDO/ETH0TXD2/CSIH0SO
	TBD	TBD	158			P24_2/MEMC0AD18/INTP2/ CSIH0CSS2/ETH0RXD1
ENCA1AIN	TBD	TBD	61	Input	ENCA1 encoder input (phase A)	P1_10/TAUA0I10/TAUA0O10/IISA2WS/ CSIH0CSS5/URTE2RX/ETH0MDC
	TBD	TBD	166			P24_8/MEMC0AD24/INTP8/ CSIH1CSS0/ETH0TXD3
ENCA1BIN	TBD	TBD	62	Input	ENCA1 encoder input (phase B)	P1_11/TAUA0I11/TAUA0O11/ IISA2SDI/IISA2SDO/CSIH0CSS6/ ETH0CRSDV/URTE2TX
	TBD	TBD	167			P24_9/MEMC0AD25/INTP9/ CSIH1CSS1/ETH0TXEN

Table 2-42 List of pin functions in alphabetical order (7/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
ENCA1TIN0	TBD	TBD	64	Input	ENCA1 capture trigger input	P1_13/TAUA0I13/TAUA0O13/ PCM0CLK/IICB1SCL
	TBD	TBD	169			P24_11/MEMC0AD27/INTP11/ CSIH1CSS3/ETH0REFCLK/ ETH0TXER
ENCA1TIN1	TBD	TBD	65			P1_14/TAUA0I14/TAUA0O14/ PCM0SEN/URTE3TX/IICB3SDA
	TBD	TBD	170			P24_12/MEMC0D28/INTP12/ CSIH1CSS4/ETH0MDI/ETH0MDO
ENCA1ZIN	TBD	TBD	63	Input	ENCA0 encoder input (phase Z)	P1_12/TAUA0I12/TAUA0O12/INTP14/ IISA3SDO/CSIH0CSS7/ETH0MDI/ ETH0MDO
	TBD	TBD	168			P24_10/MEMC0AD26/INTP10/ CSIH1CSS2/ETH0MDC
ETH0COL	TBD	TBD	73	Input	Ethernet connection pins	P3_4/TAUB2I6/TAUB2O6/INTP15/ IISA3SDO/KR0I4/CSIH1CSS4
	TBD	TBD	129			P21_0/$\overline{\text{MEMC0BEN3}}$/MEMC0DQM3
ETH0CRSDV	TBD	TBD	62	Input		P1_11/TAUA0I11/TAUA0O11/ IISA2SDI/IISA2SDO/ENCA1BIN/ CSIH0CSS6/URTE2TX
	TBD	TBD	130			P21_1/$\overline{\text{MEMC0BEN2}}$/MEMC0DQM2
ETH0MDC	TBD	TBD	61	Output		P1_10/TAUA0I10/TAUA0O10/IISA2WS/ ENCA1AIN/CSIH0CSS5/URTE2RX
	TBD	TBD	168			P24_10/MEMC0AD26/INTP10/ CSIH1CSS2/ENCA1ZIN
ETH0MDI	TBD	TBD	63	Input		P1_12/TAUA0I12/TAUA0O12/INTP14/ IISA3SDO/ENCA1ZIN/CSIH0CSS7/ ETH0MDO
	TBD	TBD	170			P24_12/MEMC0D28/INTP12/ CSIH1CSS4/ENCA1TIN1/ETH0MDO
ETH0MDO	TBD	TBD	63	Output		P1_12/TAUA0I12/TAUA0O12/INTP14/ IISA3SDO/ENCA1ZIN/CSIH0CSS7/ ETH0MDI
	TBD	TBD	170			P24_12/MEMC0D28/INTP12/ CSIH1CSS4/ENCA1TIN1/ETH0MDI
ETH0REFCLK	TBD	TBD	72	Input		P3_3/TAUB2I5/TAUB2O5/IISAACK/ KR0I3/CSIH1CSS3/ETH0TXER
	TBD	TBD	169			P24_11/MEMC0AD27/INTP11/ CSIH1CSS3/ENCA1TIN0/ETH0TXER
ETH0RXCLK	TBD	TBD	76	Input		P3_7/TAUB2I10/TAUB2O10/IISA3SDI/ IISA3SDO/KR0I7/CSIH1CSS7
	TBD	TBD	173			P24_15/MEMC0D31/INTP15/ CSIH1CSS7
ETH0RXD0	TBD	TBD	50	Input		P1_1/TAUA0I1/TAUA0O1/IISA0SCK/ $\overline{\text{CSIG4SSI}}$/URTE1TX/CSIH0CSS1
	TBD	TBD	155			P24_1/MEMC0AD17/INTP1/ CSIH0CSS1/ENCA0BIN

Table 2-42 List of pin functions in alphabetical order (8/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
ETH0RXD1	TBD	TBD	51	Input	Ethernet connection pins	P1_2/TAUA0I2/TAUA0O2/IISA0WS/ CSIG4SO/CSIH0CSS2
	TBD	TBD	158			P24_2/MEMC0AD18/INTP2/ CSIH0CSS2/ENCA0ZIN
ETH0RXD2	TBD	TBD	52			P1_3/TAUA0I3/TAUA0O3/IISA0SDI/ IISA0SDO/CSIG4SI/CSIH0CSS3
	TBD	TBD	159			P24_3/MEMC0AD19/INTP3/ CSIH0CSS3/ENCA0TIN0
ETH0RXD3	TBD	TBD	53			P1_4/TAUA0I4/TAUA0O4/INTP12/ IISA1SDO/CSIG4SC/CSIH0CSS4
	TBD	TBD	160			P24_4/MEMC0AD20/INTP4/ CSIH0CSS4/ENCA0TIN1
ETH0RXDV	TBD	TBD	75	Input		P3_6/TAUB2I9/TAUB2O9/IISA3WS/ KR0I6/CSIH1CSS6
	TBD	TBD	172			P24_14/MEMC0D30/INTP14/ CSIH1CSS6
ETH0RXER	TBD	TBD	49	Input		P1_0/TAUA0I0/TAUA0O0/IISAACK/ URTE1RX/CSIH0CSS0
	TBD	TBD	154			P24_0/MEMC0AD16/INTP0/ CSIH0CSS0/ENCA0AIN
ETH0TXCLK	TBD	TBD	74	Input		P3_5/TAUB2I7/TAUB2O7/IISA3SCK/ KR0I5/CSIH1CSS5
	TBD	TBD	171			P24_13/MEMC0D29/INTP13/ CSIH1CSS5
ETH0TXD0	TBD	TBD	54	Output		P1_5/TAUA0I5/TAUA0O5/IISA1SCK/ ENCA0AIN/$\overline{\text{CSIH0SSI}}$
	TBD	TBD	161			P24_5/MEMC0AD21/INTP5/ CSIH0CSS5
ETH0TXD1	TBD	TBD	55			P1_6/TAUA0I6/TAUA0O6/IISA1WS/ ENCA0BIN/CSIH0SI
	TBD	TBD	162			P24_6/MEMC0AD22/INTP6/ CSIH0CSS6
ETH0TXD2	TBD	TBD	58			P1_7/TAUA0I7/TAUA0O7/IISA1SDI/ IISA1SDO/ENCA0ZIN/CSIH0SO
	TBD	TBD	165			P24_7/MEMC0AD23/INTP7/ CSIH0CSS7
ETH0TXD3	TBD	TBD	59	Output		P1_8/TAUA0I8/TAUA0O8/INTP13/ IISA2SDO/ENCA0TIN0/CSIH0RYI/ CSIH0RYO
	TBD	TBD	166			P24_8/MEMC0AD24/INTP8/ CSIH1CSS0/ENCA1AIN
ETH0TXEN	TBD	TBD	60	Output		P1_9/TAUA0I9/TAUA0O9/IISA2SCK/ ENCA0TIN1/CSIH0SC
	TBD	TBD	167			P24_9/MEMC0AD25/INTP9/ CSIH1CSS1/ENCA1BIN

Table 2-42 List of pin functions in alphabetical order (9/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
ETH0TXER	TBD	TBD	72	Output	Ethernet connection pins	P3_3/TAUB2I5/TAUB2O5/IISAACK/ KR0I3/CSIH1CSS3/ETH0REFCLK
	TBD	TBD	169			P24_11/MEMC0AD27/INTP11/ CSIH1CSS3/ENCA1TIN0/ ETH0REFCLK
FCN0RX	TBD	TBD	42	Input	CAN0 receive data input	P0_5/$\overline{\text{IEBB0TX}}$/INTP9/URTE0RX/ CSIH1SO/IICB2SCL
	TBD	TBD	144			P21_13/$\overline{\text{MEMC0HLDRQ}}$/ $\overline{\text{MEMC0DSTB}}$/MEMC0CKE/CSIG4SO/ IICB2SDA
FCN0TX	TBD	TBD	41	Output	CAN0 transmit data output	P0_4/$\overline{\text{IEBB0RX}}$/INTP8/CSIH1SI/ URTE0TX/IICB2SDA
	TBD	TBD	145			P21_14/$\overline{\text{MEMC0HLDAK}}$/URTE0RX/ CSIG4SI/IICB2SCL
FCN1RX	TBD	TBD	44	Input	CAN1 receive data input	P0_7/INTP11/CSIH1SC/URTE10RX
	TBD	TBD	141			P21_10/$\overline{\text{MEMC0CS3}}$
FCN1TX	TBD	TBD	43	Output	CAN1 transmit data output	P0_6/INTP10/CSIH1RYI/CSIH1RYO/ $\overline{\text{CSIH1SSI}}$/URTE10TX
	TBD	TBD	142			P21_11/$\overline{\text{MEMC0CS4}}$
FLMD0	TBD	TBD	36	Input	Setting flash programming mode	–
FLMD1	TBD	TBD	38			P0_1/TAUJ0I1/TAUJ0O1/INTP5/ CSIG0SO/IICB0SCL
<R> FVDD	TBD	TBD	33	–	Positive power supply for flash module	–
<R> IC	TBD	TBD	21	–	Internally connected Be sure to input a low-level voltage to the IC pin.	–
$\overline{\text{IEBB0RX}}$	TBD	TBD	41	Input	IEBus receive data input	P0_4/FCN0TX/INTP8/CSIH1SI/ URTE0TX/IICB2SDA
	TBD	TBD	133			P21_4/$\overline{\text{MEMC0WR}}$/URTE2TX/ $\overline{\text{CSIG0SSI}}$/$\overline{\text{MEMC0WE}}$/IICB0SDA
$\overline{\text{IEBB0TX}}$	TBD	TBD	42	Output	IEBus transmit data output	P0_5/FCN0RX/INTP9/URTE0RX/ CSIH1SO/IICB2SCL
	TBD	TBD	134			P21_5/$\overline{\text{MEMC0RD}}$/URTE2RX/ CSIG0SO/IICB0SCL
IICB0SCL	TBD	TBD	38	I/O	IICB0 serial clock I/O	P0_1/TAUJ0I1/TAUJ0O1/INTP5/ CSIG0SO/FLMD1
	TBD	TBD	68			P3_1/TAUB2I2/TAUB2O2/PCM1SEN/ KR0I1/CSIH1CSS1
	TBD	TBD	134			P21_5/$\overline{\text{MEMC0RD}}$/URTE2RX/ $\overline{\text{IEBB0TX}}$/CSIG0SO
IICB0SDA	TBD	TBD	37	I/O	IICB0 data I/O	P0_0/TAUJ0I0/TAUJ0O0/INTP4/ $\overline{\text{CSIG0SSI}}$/$\overline{\text{RESETOUT}}$
	TBD	TBD	67			P3_0/TAUB2I1/TAUB2O1/PCM1CLK/ KR0I0/CSIH1CSS0
	TBD	TBD	133			P21_4/$\overline{\text{MEMC0WR}}$/$\overline{\text{IEBB0RX}}$/ URTE2TX/$\overline{\text{CSIG0SSI}}$/$\overline{\text{MEMC0WE}}$

Table 2-42 List of pin functions in alphabetical order (10/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
IICB1SCL	TBD	TBD	40	I/O	IICB1 serial clock I/O	P0_3/TAUJ0I3/TAUJ0O3/INTP7/CSIG0SC
	TBD	TBD	64			P1_13/TAUA0I13/TAUA0O13/PCM0CLK/ENCA1TIN0
	TBD	TBD	138			P21_7/$\overline{\text{MEMC0WAIT}}$/URTE1RX/CSIG0SC
IICB1SDA	TBD	TBD	39	I/O	IICB1 data I/O	P0_2/TAUJ0I2/TAUJ0O2/INTP6/CSIG0SI
	TBD	TBD	69			P3_2/TAUB2I3/TAUB2O3/PCM1SI/PCM1SO/KR0I2/CSIH1CSS2
	TBD	TBD	137			P21_6/MEMC0CLK/URTE1TX/CSIG0SI
IICB2SCL	TBD	TBD	42	I/O	IICB2 serial clock I/O	P0_5/FCN0RX/$\overline{\text{IEBB0TX}}$/INTP9/URTE0RX/CSIH1SO
	TBD	TBD	145			P21_14/$\overline{\text{MEMC0HLDAK}}$/URTE0RX/FCN0TX/CSIG4SI
IICB2SDA	TBD	TBD	41	I/O	IICB2 data I/O	P0_4/$\overline{\text{IEBB0RX}}$/FCN0TX/INTP8/CSIH1SI/URTE0TX
	TBD	TBD	144			P21_13/$\overline{\text{MEMC0HLDRQ}}$/$\overline{\text{MEMC0DSTB}}$/FCN0RX/MEMC0CKE/CSIG4SO
IICB3SCL	TBD	TBD	66	I/O	IICB3 serial clock I/O	P1_15/TAUA0I15/TAUA0O15/PCM0SI/PCM0SO/URTE3RX
	TBD	TBD	140			P21_9/$\overline{\text{MEMC0CS2}}$/$\overline{\text{MEMC0SDCAS}}$
IICB3SDA	TBD	TBD	65	I/O	IICB3 data I/O	P1_14/TAUA0I14/TAUA0O14/PCM0SEN/ENCA1TIN1/URTE3TX
	TBD	TBD	139			P21_8/$\overline{\text{MEMC0SDRAS}}$
IISA0SCK	TBD	TBD	50	I/O	IISA0 serial clock I/O	P1_1/TAUA0I1/TAUA0O1/$\overline{\text{CSIG4SSI}}$/URTE1TX/ETH0RXD0/CSIH0CSS1
	TBD	TBD	117			P26_12/INTP12/MEMC0A12/TAUB2I12/TAUB2O12/CSIH2CSS7
	TBD	TBD	127			P27_4/INTP4/MEMC0A20/URTE10TX
	TBD	TBD	175			P25_1/MEMC0AD1/TAUA0I1/TAUA0O1/CSIH0SI
IISA0SDI	TBD	TBD	1	Input	IISA0 serial data input	P25_3/MEMC0AD3/TAUA0I3/TAUA0O3/IISA0SDO/CSIH0RYI/CSIH0RYO
	TBD	TBD	52			P1_3/TAUA0I3/TAUA0O3/IISA0SDO/CSIG4SI/ETH0RXD2/CSIH0CSS3
	TBD	TBD	126			P27_3/INTP3/MEMC0A19/IISA0SDO/CSIH1CSS3

Table 2-42 List of pin functions in alphabetical order (11/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
IISA0SDO	TBD	TBD	1	Output	IISA0 serial data output	P25_3/MEMC0AD3/TAUA0I3/ TAUA0O3/IISA0SDI/CSIH0RYI/ CSIH0RYO
	TBD	TBD	52			P1_3/TAUA0I3/TAUA0O3/IISA0SDI/ CSIG4SI/ETH0RXD2/CSIH0CSS3
	TBD	TBD	126			P27_3/INTP3/MEMC0A19/IISA0SDI/ CSIH1CSS3
	TBD	TBD	174			P25_0/MEMC0AD0/TAUA0I0/ TAUA0O0/IISAACK/$\overline{\text{CSIH0SSI}}$
IISA0WS	TBD	TBD	51	I/O	IISA0 word select signal I/O	P1_2/TAUA0I2/TAUA0O2/CSIG4SO/ ETH0RXD1/CSIH0CSS2
	TBD	TBD	118			P26_13/INTP13/MEMC0A13/ TAUB2I13/TAUB2O13
	TBD	TBD	128			P27_5/INTP5/MEMC0A21/URTE10RX
	TBD	TBD	176			P25_2/MEMC0AD2/TAUA0I2/ TAUA0O2/CSIH0SO
IISA1SCK	TBD	TBD	3	I/O	IISA1 serial clock I/O	P25_5/MEMC0AD5/TAUA0I5/ TAUA0O5
	TBD	TBD	54			P1_5/TAUA0I5/TAUA0O5/ENCA0AIN/ ETH0TXD0/$\overline{\text{CSIH0SSI}}$
	TBD	TBD	152			P28_0/INTP6/MEMC0A22/MEMC0A24
IISA1SDI	TBD	TBD	5	Input	IISA1 serial data input	P25_7/MEMC0AD7/TAUA0I7/ TAUA0O7/IISA1SDO
	TBD	TBD	58			P1_7/TAUA0I7/TAUA0O7/IISA1SDO/ ENCA0ZIN/ETH0TXD2/CSIH0SO
	TBD	TBD	123			P27_2/INTP2/MEMC0A18/IISA1SDO/ CSIH1CSS2
IISA1SDO	TBD	TBD	2	Output	IISA1 serial data output	P25_4/MEMC0AD4/TAUA0I4/ TAUA0O4/CSIH0SC
	TBD	TBD	5			P25_7/MEMC0AD7/TAUA0I7/ TAUA0O7/IISA1SDI
	TBD	TBD	53			P1_4/TAUA0I4/TAUA0O4/INTP12/ CSIG4SC/ETH0RXD3/CSIH0CSS4
	TBD	TBD	58			P1_7/TAUA0I7/TAUA0O7/IISA1SDI/ ENCA0ZIN/ETH0TXD2/CSIH0SO
	TBD	TBD	123			P27_2/INTP2/MEMC0A18/IISA1SDI/ CSIH1CSS2
IISA1WS	TBD	TBD	4	I/O	IISA1 word select signal I/O	P25_6/MEMC0AD6/TAUA0I6/ TAUA0O6
	TBD	TBD	55			P1_6/TAUA0I6/TAUA0O6/ENCA0BIN/ ETH0TXD1/CSIH0SI
	TBD	TBD	153			P28_1/INTP7/MEMC0A23/MEMC0A25
IISA2SCK	TBD	TBD	9	I/O	IISA2 serial clock I/O	P25_9/MEMC0AD9/TAUA0I9/ TAUA0O9/CSIH1SI
	TBD	TBD	60			P1_9/TAUA0I9/TAUA0O9/ENCA0TIN1/ ETH0TXEN/CSIH0SC
	TBD	TBD	115			P26_10/INTP10/MEMC0A10/ TAUB2I10/TAUB2O10/CSIH2CSS5

Table 2-42 List of pin functions in alphabetical order (12/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
IISA2SDI	TBD	TBD	11	Input	IISA2 serial data input	P25_11/MEMC0AD11/TAUA0I11/ TAUA0O11/IISA2SDO/CSIH1RYI/ CSIH1RYO
	TBD	TBD	62			P1_11/TAUA0I11/TAUA0O11/ IISA2SDO/ENCA1BIN/CSIH0CSS6/ ETH0CRSDV/URTE2TX
	TBD	TBD	122			P27_1/INTP1/MEMC0A17/IISA2SDO/ CSIH1CSS1
IISA2SDO	TBD	TBD	8	Output	IISA2 serial data output	P25_8/MEMC0AD8/TAUA0I8/ TAUA0O8/$\overline{\text{CSIH1SSI}}$
	TBD	TBD	11			P25_11/MEMC0AD11/TAUA0I11/ TAUA0O11/IISA2SDI/CSIH1RYI/ CSIH1RYO
	TBD	TBD	59			P1_8/TAUA0I8/TAUA0O8/INTP13/ ENCA0TIN0/ETH0TXD3/CSIH0RYI/ CSIH0RYO
	TBD	TBD	62			P1_11/TAUA0I11/TAUA0O11/ IISA2SDI/ENCA1BIN/CSIH0CSS6/ ETH0CRSDV/URTE2TX
	TBD	TBD	122			P27_1/INTP1/MEMC0A17/IISA2SDI/ CSIH1CSS1
IISA2WS	TBD	TBD	10	I/O	IISA2 word select signal I/O	P25_10/MEMC0AD10/TAUA0I10/ TAUA0O10/CSIH1SO
	TBD	TBD	61			P1_10/TAUA0I10/TAUA0O10/ ENCA1AIN/CSIH0CSS5/URTE2RX/ ETH0MDC
	TBD	TBD	116			P26_11/INTP11/MEMC0A11/ TAUB2I11/TAUB2O11/CSIH2CSS6
IISA3SCK	TBD	TBD	13	I/O	IISA3 serial clock I/O	P25_13/MEMC0AD13/TAUA0I13/ TAUA0O13
	TBD	TBD	74			P3_5/TAUB2I7/TAUB2O7/KR0I5/ CSIH1CSS5/ETH0TXCLK
	TBD	TBD	111			P26_8/INTP8/MEMC0A8/TAUB2I8/ TAUB2O8/CSIH2CSS3
IISA3SDI	TBD	TBD	15	Input	IISA3 serial data input	P25_15/MEMC0AD15/TAUA0I15/ TAUA0O15/IISA3SDO
	TBD	TBD	76			P3_7/TAUB2I10/TAUB2O10/ IISA3SDO/KR0I7/CSIH1CSS7/ ETH0RXCLK
	TBD	TBD	121			P27_0/INTP0/MEMC0A16/IISA3SDO/ CSIH1CSS0

Table 2-42 List of pin functions in alphabetical order (13/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
IISA3SDO	TBD	TBD	12	Output	IISA3 serial data output	P25_12/MEMC0AD12/TAUA0I12/ TAUA0O12/CSIH1SC
	TBD	TBD	15			P25_15/MEMC0AD15/TAUA0I15/ TAUA0O15/IISA3SDI
	TBD	TBD	63			P1_12/TAUA0I12/TAUA0O12/INTP14/ ENCA1ZIN/CSIH0CSS7/ETH0MDI/ ETH0MDO
	TBD	TBD	73			P3_4/TAUB2I6/TAUB2O6/INTP15/ KR0I4/CSIH1CSS4/ETH0COL
	TBD	TBD	76			P3_7/TAUB2I10/TAUB2O10/IISA3SDI/ KR0I7/CSIH1CSS7/ETH0RXCLK
	TBD	TBD	121			P27_0/INTP0/MEMC0A16/IISA3SDI/ CSIH1CSS0
IISA3WS	TBD	TBD	14	I/O	IISA3 word select signal I/O	P25_14/MEMC0AD14/TAUA0I14/ TAUA0O14
	TBD	TBD	75			P3_6/TAUB2I9/TAUB2O9/KR0I6/ CSIH1CSS6/ETH0RXDV
	TBD	TBD	114			P26_9/INTP9/MEMC0A9/TAUB2I9/ TAUB2O9/CSIH2CSS4
IISA4SCK	TBD	TBD	104	I/O	IISA4 serial clock I/O	P26_1/KR0I1/MEMC0A1/TAUB2I1/ TAUB2O1/CSIH2SI
IISA4SDI	TBD	TBD	106	Input	IISA4 serial data input	P26_3/KR0I3/MEMC0A3/TAUB2I3/ TAUB2O3/IISA4SDO/CSIH2RYI/ CSIH2RYO
	TBD	TBD	120			P26_15/INTP15/MEMC0A15/ TAUB2I15/TAUB2O15/IISA4SDO
IISA4SDO	TBD	TBD	103	Output	IISA4 serial data output	P26_0/KR0I0/MEMC0A0/TAUB2I0/ TAUB2O0/$\overline{\text{CSIH2SSI}}$
	TBD	TBD	106			P26_3/KR0I3/MEMC0A3/TAUB2I3/ TAUB2O3/IISA4SDI/CSIH2RYI/ CSIH2RYO
	TBD	TBD	120			P26_15/INTP15/MEMC0A15/ TAUB2I15/TAUB2O15/IISA4SDI
IISA4WS	TBD	TBD	105	I/O	IISA4 word select signal I/O	P26_2/KR0I2/MEMC0A2/TAUB2I2/ TAUB2O2/CSIH2SO
IISA5SCK	TBD	TBD	108	I/O	IISA5 serial clock I/O	P26_5/KR0I5/MEMC0A5/TAUB2I5/ TAUB2O5/CSIH2CSS0
IISA5SDI	TBD	TBD	110	Input	IISA5 serial data input	P26_7/KR0I7/MEMC0A7/TAUB2I7/ TAUB2O7/IISA5SDO/CSIH2CSS2
	TBD	TBD	119			P26_14/INTP14/MEMC0A14/ TAUB2I14/TAUB2O14/IISA5SDO
IISA5SDO	TBD	TBD	107	Output	IISA5 serial data output	P26_4/KR0I4/MEMC0A4/TAUB2I4/ TAUB2O4/CSIH2SC
	TBD	TBD	110			P26_7/KR0I7/MEMC0A7/TAUB2I7/ TAUB2O7/IISA5SDI/CSIH2CSS2
	TBD	TBD	119			P26_14/INTP14/MEMC0A14/ TAUB2I14/TAUB2O14/IISA5SDI
IISA5WS	TBD	TBD	109	I/O	IISA5 word select signal I/O	P26_6/KR0I6/MEMC0A6/TAUB2I6/ TAUB2O6/CSIH2CSS1

Table 2-42 List of pin functions in alphabetical order (14/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
IISAACK	TBD	TBD	49	I/O	IISAn audio system clock I/O	P1_0/TAUA0I0/TAUA0O0/URTE1RX/ETH0RXER/CSIH0CSS0
	TBD	TBD	72			P3_3/TAUB2I5/TAUB2O5/KR0I3/CSIH1CSS3/ETH0REFCLK/ETH0TXER
	TBD	TBD	174	Input		P25_0/MEMC0AD0/TAUA0I0/TAUA0O0/IISA0SDO/$\overline{\text{CSIH0SSI}}$
INTP0	TBD	TBD	22	Input	External maskable interrupt request input	JP0_0/TAUJ0I0/TAUJ0O0/DCUTDI
	TBD	TBD	121			P27_0/MEMC0A16/IISA3SDI/IISA3SDO/CSIH1CSS0
	TBD	TBD	154			P24_0/MEMC0AD16/CSIH0CSS0/ENCA0AIN/ETH0RXER
INTP1	TBD	TBD	23			JP0_1/TAUJ0I1/TAUJ0O1/DCUTDO
	TBD	TBD	122			P27_1/MEMC0A17/IISA2SDI/IISA2SDO/CSIH1CSS1
	TBD	TBD	155			P24_1/MEMC0AD17/CSIH0CSS1/ENCA0BIN/ETH0RXD0
INTP2	TBD	TBD	24			JP0_2/TAUJ0I2/TAUJ0O2/DCUTCK
	TBD	TBD	123			P27_2/MEMC0A18/IISA1SDI/IISA1SDO/CSIH1CSS2
	TBD	TBD	158			P24_2/MEMC0AD18/CSIH0CSS2/ENCA0ZIN/ETH0RXD1
INTP3	TBD	TBD	25			JP0_3/TAUJ0I3/TAUJ0O3/DCUTMS
	TBD	TBD	126			P27_3/MEMC0A19/IISA0SDI/IISA0SDO/CSIH1CSS3
	TBD	TBD	159			P24_3/MEMC0AD19/CSIH0CSS3/ENCA0TIN0/ETH0RXD2
INTP4	TBD	TBD	37			P0_0/TAUJ0I0/TAUJ0O0/$\overline{\text{CSIG0SSI}}$/IICB0SDA/$\overline{\text{RESETOUT}}$
	TBD	TBD	127			P27_4/MEMC0A20/IISA0SCK/URTE10TX
	TBD	TBD	160			P24_4/MEMC0AD20/CSIH0CSS4/ENCA0TIN1/ETH0RXD3
INTP5	TBD	TBD	38			P0_1/TAUJ0I1/TAUJ0O1/CSIG0SO/IICB0SCL/FLMD1
	TBD	TBD	128			P27_5/MEMC0A21/IISA0WS/URTE10RX
	TBD	TBD	161			P24_5/MEMC0AD21/CSIH0CSS5/ETH0TXD0
INTP6	TBD	TBD	39			P0_2/TAUJ0I2/TAUJ0O2/CSIG0SI/IICB1SDA
	TBD	TBD	152			P28_0/MEMC0A22/MEMC0A24/IISA1SCK
	TBD	TBD	162			P24_6/MEMC0AD22/CSIH0CSS6/ETH0TXD1

Table 2-42 List of pin functions in alphabetical order (15/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
INTP7	TBD	TBD	40	Input	External maskable interrupt request input	P0_3/TAUJ0I3/TAUJ0O3/CSIG0SC/ IICB1SCL
	TBD	TBD	153			P28_1/MEMC0A23/MEMC0A25/ IISA1WS
	TBD	TBD	165			P24_7/MEMC0AD23/CSIH0CSS7/ ETH0TXD2
INTP8	TBD	TBD	41			P0_4/$\overline{\text{IEBB0RX}}$/FCN0TX/CSIH1SI/ URTE0TX/IICB2SDA
	TBD	TBD	111			P26_8/MEMC0A8/TAUB2I8/TAUB2O8/ IISA3SCK/CSIH2CSS3
	TBD	TBD	166			P24_8/MEMC0AD24/CSIH1CSS0/ ENCA1AIN/ETH0TXD3
INTP9	TBD	TBD	42			P0_5/FCN0RX/$\overline{\text{IEBB0TX}}$/URTE0RX/ CSIH1SO/IICB2SCL
	TBD	TBD	114			P26_9/MEMC0A9/TAUB2I9/TAUB2O9/ IISA3WS/CSIH2CSS4
	TBD	TBD	167			P24_9/MEMC0AD25/CSIH1CSS1/ ENCA1BIN/ETH0TXEN
INTP10	TBD	TBD	43			P0_6/FCN1TX/CSIH1RYI/CSIH1RYO/ $\overline{\text{CSIH1SSI}}$/URTE10TX
	TBD	TBD	115			P26_10/MEMC0A10/TAUB2I10/ TAUB2O10/IISA2SCK/CSIH2CSS5
	TBD	TBD	168			P24_10/MEMC0AD26/CSIH1CSS2/ ENCA1ZIN/ETH0MDC
INTP11	TBD	TBD	44			P0_7/FCN1RX/CSIH1SC/URTE10RX
	TBD	TBD	116			P26_11/MEMC0A11/TAUB2I11/ TAUB2O11/IISA2WS/CSIH2CSS6
	TBD	TBD	169			P24_11/MEMC0AD27/CSIH1CSS3/ ENCA1TIN0/ETH0REFCLK/ ETH0TXER
INTP12	TBD	TBD	53			P1_4/TAUA0I4/TAUA0O4/IISA1SDO/ CSIG4SC/ETH0RXD3/CSIH0CSS4
	TBD	TBD	117			P26_12/MEMC0A12/TAUB2I12/ TAUB2O12/IISA0SCK/CSIH2CSS7
	TBD	TBD	170			P24_12/MEMC0D28/CSIH1CSS4/ ENCA1TIN1/ETH0MDI/ETH0MDO
INTP13	TBD	TBD	59			P1_8/TAUA0I8/TAUA0O8/IISA2SDO/ ENCA0TIN0/ETH0TXD3/CSIH0RYI/ CSIH0RYO
	TBD	TBD	118			P26_13/MEMC0A13/TAUB2I13/ TAUB2O13/IISA0WS
	TBD	TBD	171			P24_13/MEMC0D29/CSIH1CSS5/ ETH0TXCLK

Table 2-42 List of pin functions in alphabetical order (16/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
INTP14	TBD	TBD	63	Input	External maskable interrupt request input	P1_12/TAUA0I12/TAUA0O12/ IISA3SDO/ENCA1ZIN/CSIH0CSS7/ ETH0MDI/ETH0MDO
	TBD	TBD	119			P26_14/MEMC0A14/TAUB2I14/ TAUB2O14/IISA5SDI/IISA5SDO
	TBD	TBD	172			P24_14/MEMC0D30/CSIH1CSS6/ ETH0RXDV
INTP15	TBD	TBD	73			P3_4/TAUB2I6/TAUB2O6/IISA3SDO/ KR0I4/CSIH1CSS4/ETH0COL
	TBD	TBD	120			P26_15/MEMC0A15/TAUB2I15/ TAUB2O15/IISA4SDI/IISA4SDO
	TBD	TBD	173			P24_15/MEMC0D31/CSIH1CSS7/ ETH0RXCLK
KR0I0	TBD	TBD	67	Input	Key interrupt input	P3_0/TAUB2I1/TAUB2O1/PCM1CLK/ CSIH1CSS0/IICB0SDA
	TBD	TBD	103			P26_0/MEMC0A0/TAUB2I0/TAUB2O0/ IISA4SDO/$\overline{\text{CSIH2SSI}}$
KR0I1	TBD	TBD	68			P3_1/TAUB2I2/TAUB2O2/PCM1SEN/ CSIH1CSS1/IICB0SCL
	TBD	TBD	104			P26_1/MEMC0A1/TAUB2I1/TAUB2O1/ IISA4SCK/CSIH2SI
KR0I2	TBD	TBD	69			P3_2/TAUB2I3/TAUB2O3/PCM1SI/ PCM1SO/CSIH1CSS2/IICB1SDA
	TBD	TBD	105			P26_2/MEMC0A2/TAUB2I2/TAUB2O2/ IISA4WS/CSIH2SO
KR0I3	TBD	TBD	72			P3_3/TAUB2I5/TAUB2O5/IISAACK/ CSIH1CSS3/ETH0REFCLK/ ETH0TXER
	TBD	TBD	106			P26_3/MEMC0A3/TAUB2I3/TAUB2O3/ IISA4SDI/IISA4SDO/CSIH2RYI/ CSIH2RYO
KR0I4	TBD	TBD	73			P3_4/TAUB2I6/TAUB2O6/INTP15/ IISA3SDO/CSIH1CSS4/ETH0COL
	TBD	TBD	107			P26_4/MEMC0A4/TAUB2I4/TAUB2O4/ IISA5SDO/CSIH2SC
KR0I5	TBD	TBD	74			P3_5/TAUB2I7/TAUB2O7/IISA3SCK/ CSIH1CSS5/ETH0TXCLK
	TBD	TBD	108			P26_5/MEMC0A5/TAUB2I5/TAUB2O5/ IISA5SCK/CSIH2CSS0
KR0I6	TBD	TBD	75			P3_6/TAUB2I9/TAUB2O9/IISA3WS/ CSIH1CSS6/ETH0RXDV
	TBD	TBD	109			P26_6/MEMC0A6/TAUB2I6/TAUB2O6/ IISA5WS/CSIH2CSS1
KR0I7	TBD	TBD	76			P3_7/TAUB2I10/TAUB2O10/IISA3SDI/ IISA3SDO/CSIH1CSS7/ETH0RXCLK
	TBD	TBD	110			P26_7/MEMC0A7/TAUB2I7/TAUB2O7/ IISA5SDI/IISA5SDO/CSIH2CSS2

Table 2-42 List of pin functions in alphabetical order (17/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
MEMC0A0	TBD	TBD	103	Output	Address bus for external memory	P26_0/KR0I0/TAUB2I0/TAUB2O0/ IISA4SDO/$\overline{\text{CSIH2SSI}}$
MEMC0A1	TBD	TBD	104			P26_1/KR0I1/TAUB2I1/TAUB2O1/ IISA4SCK/CSIH2SI
MEMC0A2	TBD	TBD	105			P26_2/KR0I2/TAUB2I2/TAUB2O2/ IISA4WS/CSIH2SO
MEMC0A3	TBD	TBD	106			P26_3/KR0I3/TAUB2I3/TAUB2O3/ IISA4SDI/IISA4SDO/CSIH2RYI/ CSIH2RYO
MEMC0A4	TBD	TBD	107			P26_4/KR0I4/TAUB2I4/TAUB2O4/ IISA5SDO/CSIH2SC
MEMC0A5	TBD	TBD	108			P26_5/KR0I5/TAUB2I5/TAUB2O5/ IISA5SCK/CSIH2CSS0
MEMC0A6	TBD	TBD	109			P26_6/KR0I6/TAUB2I6/TAUB2O6/ IISA5WS/CSIH2CSS1
MEMC0A7	TBD	TBD	110			P26_7/KR0I7/TAUB2I7/TAUB2O7/ IISA5SDI/IISA5SDO/CSIH2CSS2
MEMC0A8	TBD	TBD	111			P26_8/INTP8/TAUB2I8/TAUB2O8/ IISA3SCK/CSIH2CSS3
MEMC0A9	TBD	TBD	114			P26_9/INTP9/TAUB2I9/TAUB2O9/ IISA3WS/CSIH2CSS4
MEMC0A10	TBD	TBD	115			P26_10/INTP10/TAUB2I10/TAUB2O10/ IISA2SCK/CSIH2CSS5
MEMC0A11	TBD	TBD	116			P26_11/INTP11/TAUB2I11/TAUB2O11/ IISA2WS/CSIH2CSS6
MEMC0A12	TBD	TBD	117			P26_12/INTP12/TAUB2I12/TAUB2O12/ IISA0SCK/CSIH2CSS7
MEMC0A13	TBD	TBD	118			P26_13/INTP13/TAUB2I13/TAUB2O13/ IISA0WS
MEMC0A14	TBD	TBD	119			P26_14/INTP14/TAUB2I14/TAUB2O14/ IISA5SDI/IISA5SDO
MEMC0A15	TBD	TBD	120			P26_15/INTP15/TAUB2I15/TAUB2O15/ IISA4SDI/IISA4SDO
MEMC0A16	TBD	TBD	121			P27_0/INTP0/IISA3SDI/IISA3SDO/ CSIH1CSS0
MEMC0A17	TBD	TBD	122			P27_1/INTP1/IISA2SDI/IISA2SDO/ CSIH1CSS1
MEMC0A18	TBD	TBD	123			P27_2/INTP2/IISA1SDI/IISA1SDO/ CSIH1CSS2
MEMC0A19	TBD	TBD	126			P27_3/INTP3/IISA0SDI/IISA0SDO/ CSIH1CSS3
MEMC0A20	TBD	TBD	127			P27_4/INTP4/IISA0SCK/URTE10TX
MEMC0A21	TBD	TBD	128			P27_5/INTP5/IISA0WS/URTE10RX
MEMC0A22	TBD	TBD	152			P28_0/INTP6/MEMC0A24/IISA1SCK
MEMC0A23	TBD	TBD	153			P28_1/INTP7/MEMC0A25/IISA1WS
MEMC0A24	TBD	TBD	152			P28_0/INTP6/MEMC0A22/IISA1SCK
MEMC0A25	TBD	TBD	153			P28_1/INTP7/MEMC0A23/IISA1WS

Table 2-42 List of pin functions in alphabetical order (18/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
MEMC0AD0	TBD	TBD	174	I/O	Address bus/data bus for external memory	P25_0/TAUA0I0/TAUA0O0/IISAACK/ IISA0SDO/$\overline{\text{CSIH0SSI}}$
MEMC0AD1	TBD	TBD	175			P25_1/TAUA0I1/TAUA0O1/IISA0SCK/ CSIH0SI
MEMC0AD2	TBD	TBD	176			P25_2/TAUA0I2/TAUA0O2/IISA0WS/ CSIH0SO
MEMC0AD3	TBD	TBD	1			P25_3/TAUA0I3/TAUA0O3/IISA0SDI/ IISA0SDO/CSIH0RYI/CSIH0RYO
MEMC0AD4	TBD	TBD	2			P25_4/TAUA0I4/TAUA0O4/IISA1SDO/ CSIH0SC
MEMC0AD5	TBD	TBD	3			P25_5/TAUA0I5/TAUA0O5/IISA1SCK
MEMC0AD6	TBD	TBD	4			P25_6/TAUA0I6/TAUA0O6/IISA1WS
MEMC0AD7	TBD	TBD	5			P25_7/TAUA0I7/TAUA0O7/IISA1SDI/ IISA1SDO
MEMC0AD8	TBD	TBD	8			P25_8/TAUA0I8/TAUA0O8/IISA2SDO/ $\overline{\text{CSIH1SSI}}$
MEMC0AD9	TBD	TBD	9			P25_9/TAUA0I9/TAUA0O9/IISA2SCK/ CSIH1SI
MEMC0AD10	TBD	TBD	10			P25_10/TAUA0I10/TAUA0O10/ IISA2WS/CSIH1SO
MEMC0AD11	TBD	TBD	11			P25_11/TAUA0I11/TAUA0O11/ IISA2SDI/IISA2SDO/CSIH1RYI/ CSIH1RYO
MEMC0AD12	TBD	TBD	12			P25_12/TAUA0I12/TAUA0O12/ IISA3SDO/CSIH1SC
MEMC0AD13	TBD	TBD	13			P25_13/TAUA0I13/TAUA0O13/ IISA3SCK
MEMC0AD14	TBD	TBD	14			P25_14/TAUA0I14/TAUA0O14/ IISA3WS
MEMC0AD15	TBD	TBD	15			P25_15/TAUA0I15/TAUA0O15/ IISA3SDI/IISA3SDO
MEMC0AD16	TBD	TBD	154			P24_0/INTP0/CSIH0CSS0/ ENCA0AIN/ETH0RXER
MEMC0AD17	TBD	TBD	155			P24_1/INTP1/CSIH0CSS1/ ENCA0BIN/ETH0RXD0
MEMC0AD18	TBD	TBD	158			P24_2/INTP2/CSIH0CSS2/ENCA0ZIN/ ETH0RXD1
MEMC0AD19	TBD	TBD	159			P24_3/INTP3/CSIH0CSS3/ ENCA0TIN0/ETH0RXD2
MEMC0AD20	TBD	TBD	160			P24_4/INTP4/CSIH0CSS4/ ENCA0TIN1/ETH0RXD3
MEMC0AD21	TBD	TBD	161			P24_5/INTP5/CSIH0CSS5/ETH0TXD0
MEMC0AD22	TBD	TBD	162			P24_6/INTP6/CSIH0CSS6/ETH0TXD1
MEMC0AD23	TBD	TBD	165			P24_7/INTP7/CSIH0CSS7/ETH0TXD2
MEMC0AD24	TBD	TBD	166			P24_8/INTP8/CSIH1CSS0/ ENCA1AIN/ETH0TXD3

Table 2-42 List of pin functions in alphabetical order (19/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
MEMC0AD25	TBD	TBD	167	I/O	Address bus/data bus for external memory	P24_9/INTP9/CSIH1CSS1/ ENCA1BIN/ETH0TXEN
MEMC0AD26	TBD	TBD	168			P24_10/INTP10/CSIH1CSS2/ ENCA1ZIN/ETH0MDC
MEMC0AD27	TBD	TBD	169			P24_11/INTP11/CSIH1CSS3/ ENCA1TIN0/ETH0REFCLK/ ETH0TXER
$\overline{\text{MEMC0ASTB}}$	TBD	TBD	146	Output	Address strobe signal output for external memory	P21_15/URTE0TX/CSIG4SC
$\overline{\text{MEMC0BEN0}}$	TBD	TBD	132	Output	External data bus byte enable signal output (D0 to D7)	P21_3/MEMC0DQM0
$\overline{\text{MEMC0BEN1}}$	TBD	TBD	131		External data bus byte enable signal output (D8 to D15)	P21_2/MEMC0DQM1
$\overline{\text{MEMC0BEN2}}$	TBD	TBD	130		External data bus byte enable signal output (D16 to D23)	P21_1/MEMC0DQM2/ETH0CRSDV
$\overline{\text{MEMC0BEN3}}$	TBD	TBD	129		External data bus byte enable signal output (D24 to D31)	P21_0/MEMC0DQM3/ETH0COL
MEMC0CKE	TBD	TBD	144	Output	SDRAM clock enable output signal	P21_13/$\overline{\text{MEMC0HLDRQ}}$/ $\overline{\text{MEMC0DSTB}}$/FCN0RX/CSIG4SO/ IICB2SDA
MEMC0CLK	TBD	TBD	137	Output	Bus clock output	P21_6/URTE1TX/CSIG0SI/IICB1SDA
$\overline{\text{MEMC0CS2}}$	TBD	TBD	140	Output	Chip select signal output for external memory	P21_9/$\overline{\text{MEMC0SDCAS}}$/IICB3SCL
$\overline{\text{MEMC0CS3}}$	TBD	TBD	141			P21_10/FCN1RX
$\overline{\text{MEMC0CS4}}$	TBD	TBD	142			P21_11/FCN1TX
$\overline{\text{MEMC0CS7}}$	TBD	TBD	143			P21_12/$\overline{\text{CSIG4SSI}}$
MEMC0D28	TBD	TBD	170	I/O	Data bus for external memory	P24_12/INTP12/CSIH1CSS4/ ENCA1TIN1/ETH0MDI/ETH0MDO
MEMC0D29	TBD	TBD	171			P24_13/INTP13/CSIH1CSS5/ ETH0TXCLK
MEMC0D30	TBD	TBD	172			P24_14/INTP14/CSIH1CSS6/ ETH0RXDV
MEMC0D31	TBD	TBD	173			P24_15/INTP15/CSIH1CSS7/ ETH0RXCLK
MEMC0DQM0	TBD	TBD	132	Output	I/O mask signal output for SDRAM (D0 to D7)	P21_3/$\overline{\text{MEMC0BEN0}}$
MEMC0DQM1	TBD	TBD	131		I/O mask signal output for SDRAM (D8 to D15)	P21_2/$\overline{\text{MEMC0BEN1}}$
MEMC0DQM2	TBD	TBD	130		I/O mask signal output for SDRAM (D16 to D23)	P21_1/$\overline{\text{MEMC0BEN2}}$/ETH0CRSDV
MEMC0DQM3	TBD	TBD	129		I/O mask signal output for SDRAM (D24 to D31)	P21_0/$\overline{\text{MEMC0BEN3}}$/ETH0COL
$\overline{\text{MEMC0DSTB}}$	TBD	TBD	144	Output	External address bus latch strobe signal output	P21_13/$\overline{\text{MEMC0HLDRQ}}$/FCN0RX/ MEMC0CKE/CSIG4SO/IICB2SDA
$\overline{\text{MEMC0HLDAK}}$	TBD	TBD	145	Output	Bus hold acknowledge output	P21_14/URTE0RX/FCN0TX/CSIG4SI/ IICB2SCL
$\overline{\text{MEMC0HLDRQ}}$	TBD	TBD	144	Input	Bus hold request input	P21_13/$\overline{\text{MEMC0DSTB}}$/FCN0RX/ MEMC0CKE/CSIG4SO/IICB2SDA

Table 2-42 List of pin functions in alphabetical order (20/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
MEMC0RD	TBD	TBD	134	Output	External data bus read strobe signal output	P21_5/URTE2RX/IEBB0TX/CSIG0SO/IICB0SCL
MEMC0SDCAS	TBD	TBD	140	Output	Column address strobe signal output to SDRAM	P21_9/MEMC0CS2/IICB3SCL
MEMC0SDRAS	TBD	TBD	139	Output	Row address strobe signal output to SDRAM	P21_8/IICB3SDA
MEMC0WAIT	TBD	TBD	138	Input	External wait request input	P21_7/URTE1RX/CSIG0SC/IICB1SCL
MEMC0WE	TBD	TBD	133	Output	Write enable signal output for SDRAM	P21_4/MEMC0WR/IEBB0RX/URTE2TX/CSIG0SSI/IICB0SDA
MEMC0WR	TBD	TBD	133	Output	External data bus write strobe signal output	P21_4/IEBB0RX/URTE2TX/CSIG0SSI/MEMC0WE/IICB0SDA
MLB0VDD	TBD	TBD	151	–	Positive power supply for MediaLB	–
MLB0VSS	TBD	TBD	147	–	Ground potential for MediaLB	–
MLBA0CLK	TBD	TBD	150	Input	Clock input for MediaLB	–
MLBA0DAT	TBD	TBD	149	I/O	MediaLB data output	–
MLBA0SIG	TBD	TBD	148	I/O	MediaLB signal information I/O	–
NMI	TBD	TBD	26	Input	Non-maskable interrupt request input	JP0_5/RTCA0OUT/DCURDY
OSCVDD	TBD	TBD	30	–	Positive power supply for oscillator	–
OSCVSS	TBD	TBD	29	–	Ground potential for oscillator	–
PCM0CLK	TBD	TBD	64	I/O	PCM0 clock signal	P1_13/TAUA0I13/TAUA0O13/ENCA1TIN0/IICB1SCL
PCM0SEN	TBD	TBD	65	I/O	PCM0 frame sync signal	P1_14/TAUA0I14/TAUA0O14/ENCA1TIN1/URTE3TX/IICB3SDA
PCM0SI	TBD	TBD	66	Input	PCM0 data input	P1_15/TAUA0I15/TAUA0O15/PCM0SO/URTE3RX/IICB3SCL
PCM0SO	TBD	TBD	66	Output	PCM0 data output	P1_15/TAUA0I15/TAUA0O15/PCM0SI/URTE3RX/IICB3SCL
PCM1CLK	TBD	TBD	67	I/O	PCM1 clock signal	P3_0/TAUB2I1/TAUB2O1/KR0I0/CSIH1CSS0/IICB0SDA
PCM1SEN	TBD	TBD	68	I/O	PCM1 frame sync signal	P3_1/TAUB2I2/TAUB2O2/KR0I1/CSIH1CSS1/IICB0SCL
PCM1SI	TBD	TBD	69	Input	PCM1 data input	P3_2/TAUB2I3/TAUB2O3/PCM1SO/KR0I2/CSIH1CSS2/IICB1SDA
PCM1SO	TBD	TBD	69	Output	PCM1 data output	P3_2/TAUB2I3/TAUB2O3/PCM1SI/KR0I2/CSIH1CSS2/IICB1SDA
PTCTL1	TBD	TBD	80	Output	External power transistor control signal	–
REG0C	TBD	TBD	17	–	Connecting voltage regulator capacitor	–
REG0VDD	TBD	TBD	16	–	Voltage regulator input	–
REG0VSS	TBD	TBD	18	–	Voltage regulator ground	–
REG1VDD	TBD	TBD	77	–	Voltage regulator input	–

Table 2-42 List of pin functions in alphabetical order (21/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
REG1VSS	TBD	TBD	79	–	Voltage regulator ground	–
$\overline{\text{RESETOUT}}$	TBD	TBD	37	Output	Reset output	P0_0/TAUJ0I0/TAUJ0O0/INTP4/ $\overline{\text{CSIG0SSI}}$/IICB0SDA
$\overline{\text{RESET}}$	TBD	TBD	19	Input	External reset input	–
RTCA0OUT	TBD	TBD	26	Output	Real-time clock 1 Hz interval output	JP0_5/NMI/$\overline{\text{DCURDY}}$
TAUA0I0	TBD	TBD	49	Input	TAUA0 timer input	P1_0/TAUA0O0/IISAACK/URTE1RX/ ETH0RXER/CSIH0CSS0
	TBD	TBD	174			P25_0/MEMC0AD0/TAUA0O0/ IISAACK/IISA0SDO/$\overline{\text{CSIH0SSI}}$
TAUA0I1	TBD	TBD	50			P1_1/TAUA0O1/IISA0SCK/$\overline{\text{CSIG4SSI}}$/ URTE1TX/ETH0RXD0/CSIH0CSS1
	TBD	TBD	175			P25_1/MEMC0AD1/TAUA0O1/ IISA0SCK/CSIH0SI
TAUA0I2	TBD	TBD	51			P1_2/TAUA0O2/IISA0WS/CSIG4SO/ ETH0RXD1/CSIH0CSS2
	TBD	TBD	176			P25_2/MEMC0AD2/TAUA0O2/ IISA0WS/CSIH0SO
TAUA0I3	TBD	TBD	1			P25_3/MEMC0AD3/TAUA0O3/ IISA0SDI/IISA0SDO/CSIH0RYI/ CSIH0RYO
	TBD	TBD	52			P1_3/TAUA0O3/IISA0SDI/IISA0SDO/ CSIG4SI/ETH0RXD2/CSIH0CSS3
TAUA0I4	TBD	TBD	2			P25_4/MEMC0AD4/TAUA0O4/ IISA1SDO/CSIH0SC
	TBD	TBD	53			P1_4/TAUA0O4/INTP12/IISA1SDO/ CSIG4SC/ETH0RXD3/CSIH0CSS4
TAUA0I5	TBD	TBD	3			P25_5/MEMC0AD5/TAUA0O5/ IISA1SCK
	TBD	TBD	54			P1_5/TAUA0O5/IISA1SCK/ENCA0AIN/ ETH0TXD0/$\overline{\text{CSIH0SSI}}$
TAUA0I6	TBD	TBD	4			P25_6/MEMC0AD6/TAUA0O6/ IISA1WS
	TBD	TBD	55			P1_6/TAUA0O6/IISA1WS/ENCA0BIN/ ETH0TXD1/CSIH0SI
TAUA0I7	TBD	TBD	5			P25_7/MEMC0AD7/TAUA0O7/ IISA1SDI/IISA1SDO
	TBD	TBD	58			P1_7/TAUA0O7/IISA1SDI/IISA1SDO/ ENCA0ZIN/ETH0TXD2/CSIH0SO
TAUA0I8	TBD	TBD	8			P25_8/MEMC0AD8/TAUA0O8/ IISA2SDO/$\overline{\text{CSIH1SSI}}$
	TBD	TBD	59			P1_8/TAUA0O8/INTP13/IISA2SDO/ ENCA0TIN0/ETH0TXD3/CSIH0RYI/ CSIH0RYO
TAUA0I9	TBD	TBD	9			P25_9/MEMC0AD9/TAUA0O9/ IISA2SCK/CSIH1SI
	TBD	TBD	60			P1_9/TAUA0O9/IISA2SCK/ ENCA0TIN1/ETH0TXEN/CSIH0SC

Table 2-42 List of pin functions in alphabetical order (22/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
TAUA0I10	TBD	TBD	10	Input	TAUA0 timer input	P25_10/MEMC0AD10/TAUA0O10/ IISA2WS/CSIH1SO
	TBD	TBD	61			P1_10/TAUA0O10/IISA2WS/ ENCA1AIN/CSIH0CSS5/URTE2RX/ ETH0MDC
TAUA0I11	TBD	TBD	11			P25_11/MEMC0AD11/TAUA0O11/ IISA2SDI/IISA2SDO/CSIH1RYI/ CSIH1RYO
	TBD	TBD	62			P1_11/TAUA0O11/IISA2SDI/ IISA2SDO/ENCA1BIN/CSIH0CSS6/ ETH0CRSDV/URTE2TX
TAUA0I12	TBD	TBD	12			P25_12/MEMC0AD12/TAUA0O12/ IISA3SDO/CSIH1SC
	TBD	TBD	63			P1_12/TAUA0O12/INTP14/IISA3SDO/ ENCA1ZIN/CSIH0CSS7/ETH0MDI/ ETH0MDO
TAUA0I13	TBD	TBD	13			P25_13/MEMC0AD13/TAUA0O13/ IISA3SCK
	TBD	TBD	64			P1_13/TAUA0O13/PCM0CLK/ ENCA1TIN0/IICB1SCL
TAUA0I14	TBD	TBD	14			P25_14/MEMC0AD14/TAUA0O14/ IISA3WS
	TBD	TBD	65			P1_14/TAUA0O14/PCM0SEN/ ENCA1TIN1/URTE3TX/IICB3SDA
TAUA0I15	TBD	TBD	15			P25_15/MEMC0AD15/TAUA0O15/ IISA3SDI/IISA3SDO
	TBD	TBD	66			P1_15/TAUA0O15/PCM0SI/PCM0SO/ URTE3RX/IICB3SCL
TAUA0O0	TBD	TBD	49	Output	TAUA0 timer output	P1_0/TAUA0I0/IISAACK/URTE1RX/ ETH0RXER/CSIH0CSS0
	TBD	TBD	174			P25_0/MEMC0AD0/TAUA0I0/IISAACK/ IISA0SDO/$\overline{\text{CSIH0SSI}}$
TAUA0O1	TBD	TBD	50			P1_1/TAUA0I1/IISA0SCK/$\overline{\text{CSIG4SSI}}$/ URTE1TX/ETH0RXD0/CSIH0CSS1
	TBD	TBD	175			P25_1/MEMC0AD1/TAUA0I1/ IISA0SCK/CSIH0SI
TAUA0O2	TBD	TBD	51			P1_2/TAUA0I2/IISA0WS/CSIG4SO/ ETH0RXD1/CSIH0CSS2
	TBD	TBD	176			P25_2/MEMC0AD2/TAUA0I2/ IISA0WS/CSIH0SO
TAUA0O3	TBD	TBD	1			P25_3/MEMC0AD3/TAUA0I3/ IISA0SDI/IISA0SDO/CSIH0RYI/ CSIH0RYO
	TBD	TBD	52			P1_3/TAUA0I3/IISA0SDI/IISA0SDO/ CSIG4SI/ETH0RXD2/CSIH0CSS3
TAUA0O4	TBD	TBD	2			P25_4/MEMC0AD4/TAUA0I4/ IISA1SDO/CSIH0SC
	TBD	TBD	53			P1_4/TAUA0I4/INTP12/IISA1SDO/ CSIG4SC/ETH0RXD3/CSIH0CSS4

Table 2-42 List of pin functions in alphabetical order (23/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
TAUA0O5	TBD	TBD	3	Output	TAUA0 timer output	P25_5/MEMC0AD5/TAUA0I5/ IISA1SCK
	TBD	TBD	54			P1_5/TAUA0I5/IISA1SCK/ENCA0AIN/ ETH0TXD0/$\overline{\text{CSIH0SSI}}$
TAUA0O6	TBD	TBD	4			P25_6/MEMC0AD6/TAUA0I6/IISA1WS
	TBD	TBD	55			P1_6/TAUA0I6/IISA1WS/ENCA0BIN/ ETH0TXD1/CSIH0SI
TAUA0O7	TBD	TBD	5			P25_7/MEMC0AD7/TAUA0I7/ IISA1SDI/IISA1SDO
TAUA0O7	TBD	TBD	58			P1_7/TAUA0I7/IISA1SDI/IISA1SDO/ ENCA0ZIN/ETH0TXD2/CSIH0SO
TAUA0O8	TBD	TBD	8			P25_8/MEMC0AD8/TAUA0I8/ IISA2SDO/$\overline{\text{CSIH1SSI}}$
	TBD	TBD	59			P1_8/TAUA0I8/INTP13/IISA2SDO/ ENCA0TIN0/ETH0TXD3/CSIH0RYI/ CSIH0RYO
TAUA0O9	TBD	TBD	9			P25_9/MEMC0AD9/TAUA0I9/ IISA2SCK/CSIH1SI
	TBD	TBD	60			P1_9/TAUA0I9/IISA2SCK/ENCA0TIN1/ ETH0TXEN/CSIH0SC
TAUA0O10	TBD	TBD	10			P25_10/MEMC0AD10/TAUA0I10/ IISA2WS/CSIH1SO
	TBD	TBD	61			P1_10/TAUA0I10/IISA2WS/ ENCA1AIN/CSIH0CSS5/URTE2RX/ ETH0MDC
TAUA0O11	TBD	TBD	11			P25_11/MEMC0AD11/TAUA0I11/ IISA2SDI/IISA2SDO/CSIH1RYI/ CSIH1RYO
	TBD	TBD	62			P1_11/TAUA0I11/IISA2SDI/IISA2SDO/ ENCA1BIN/CSIH0CSS6/ ETH0CRSDV/URTE2TX
TAUA0O12	TBD	TBD	12			P25_12/MEMC0AD12/TAUA0I12/ IISA3SDO/CSIH1SC
	TBD	TBD	63			P1_12/TAUA0I12/INTP14/IISA3SDO/ ENCA1ZIN/CSIH0CSS7/ETH0MDI/ ETH0MDO
TAUA0O13	TBD	TBD	13			P25_13/MEMC0AD13/TAUA0I13/ IISA3SCK
	TBD	TBD	64			P1_13/TAUA0I13/PCM0CLK/ ENCA1TIN0/IICB1SCL
TAUA0O14	TBD	TBD	14			P25_14/MEMC0AD14/TAUA0I14/ IISA3WS
	TBD	TBD	65			P1_14/TAUA0I14/PCM0SEN/ ENCA1TIN1/URTE3TX/IICB3SDA
TAUA0O15	TBD	TBD	15			P25_15/MEMC0AD15/TAUA0I15/ IISA3SDI/IISA3SDO
	TBD	TBD	66			P1_15/TAUA0I15/PCM0SI/PCM0SO/ URTE3RX/IICB3SCL

Table 2-42 List of pin functions in alphabetical order (24/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
TAUB2I0	TBD	TBD	103	Input	TAUB2 timer input	P26_0/KR0I0/MEMC0A0/TAUB2O0/ IISA4SDO/$\overline{\text{CSIH2SSI}}$
TAUB2I1	TBD	TBD	67			P3_0/TAUB2O1/PCM1CLK/KR0I0/ CSIH1CSS0/IICB0SDA
	TBD	TBD	104			P26_1/KR0I1/MEMC0A1/TAUB2O1/ IISA4SCK/CSIH2SI
TAUB2I2	TBD	TBD	68			P3_1/TAUB2O2/PCM1SEN/KR0I1/ CSIH1CSS1/IICB0SCL
	TBD	TBD	105			P26_2/KR0I2/MEMC0A2/TAUB2O2/ IISA4WS/CSIH2SO
TAUB2I3	TBD	TBD	69			P3_2/TAUB2O3/PCM1SI/PCM1SO/ KR0I2/CSIH1CSS2/IICB1SDA
	TBD	TBD	106			P26_3/KR0I3/MEMC0A3/TAUB2O3/ IISA4SDI/IISA4SDO/CSIH2RYI/ CSIH2RYO
TAUB2I4	TBD	TBD	107			P26_4/KR0I4/MEMC0A4/TAUB2O4/ IISA5SDO/CSIH2SC
TAUB2I5	TBD	TBD	72			P3_3/TAUB2O5/IISAACK/KR0I3/ CSIH1CSS3/ETH0REFCLK/ ETH0TXER
	TBD	TBD	108			P26_5/KR0I5/MEMC0A5/TAUB2O5/ IISA5SCK/CSIH2CSS0
TAUB2I6	TBD	TBD	73			P3_4/TAUB2O6/INTP15/IISA3SDO/ KR0I4/CSIH1CSS4/ETH0COL
	TBD	TBD	109			P26_6/KR0I6/MEMC0A6/TAUB2O6/ IISA5WS/CSIH2CSS1
TAUB2I7	TBD	TBD	74			P3_5/TAUB2O7/IISA3SCK/KR0I5/ CSIH1CSS5/ETH0TXCLK
	TBD	TBD	110			P26_7/KR0I7/MEMC0A7/TAUB2O7/ IISA5SDI/IISA5SDO/CSIH2CSS2
TAUB2I8	TBD	TBD	111			P26_8/INTP8/MEMC0A8/TAUB2O8/ IISA3SCK/CSIH2CSS3
TAUB2I9	TBD	TBD	75			P3_6/TAUB2O9/IISA3WS/KR0I6/ CSIH1CSS6/ETH0RXDV
	TBD	TBD	114			P26_9/INTP9/MEMC0A9/TAUB2O9/ IISA3WS/CSIH2CSS4
TAUB2I10	TBD	TBD	76			P3_7/TAUB2O10/IISA3SDI/IISA3SDO/ KR0I7/CSIH1CSS7/ETH0RXCLK
	TBD	TBD	115			P26_10/INTP10/MEMC0A10/ TAUB2O10/IISA2SCK/CSIH2CSS5
TAUB2I11	TBD	TBD	116			P26_11/INTP11/MEMC0A11/ TAUB2O11/IISA2WS/CSIH2CSS6
TAUB2I12	TBD	TBD	117			P26_12/INTP12/MEMC0A12/ TAUB2O12/IISA0SCK/CSIH2CSS7
TAUB2I13	TBD	TBD	118			P26_13/INTP13/MEMC0A13/ TAUB2O13/IISA0WS
TAUB2I14	TBD	TBD	119			P26_14/INTP14/MEMC0A14/ TAUB2O14/IISA5SDI/IISA5SDO

Table 2-42 List of pin functions in alphabetical order (25/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
TAUB2I15	TBD	TBD	120	Input	TAUB2 timer input	P26_15/INTP15/MEMC0A15/ TAUB2O15/IISA4SDI/IISA4SDO
TAUB2O0	TBD	TBD	103	Output	TAUB2 timer output	P26_0/KR0I0/MEMC0A0/TAUB2I0/ IISA4SDO/$\overline{\text{CSIH2SSI}}$
TAUB2O1	TBD	TBD	67			P3_0/TAUB2I1/PCM1CLK/KR0I0/ CSIH1CSS0/IICB0SDA
	TBD	TBD	104			P26_1/KR0I1/MEMC0A1/TAUB2I1/ IISA4SCK/CSIH2SI
TAUB2O2	TBD	TBD	68			P3_1/TAUB2I2/PCM1SEN/KR0I1/ CSIH1CSS1/IICB0SCL
TAUB2O2	TBD	TBD	105			P26_2/KR0I2/MEMC0A2/TAUB2I2/ IISA4WS/CSIH2SO
TAUB2O3	TBD	TBD	69			P3_2/TAUB2I3/PCM1SI/PCM1SO/ KR0I2/CSIH1CSS2/IICB1SDA
	TBD	TBD	106			P26_3/KR0I3/MEMC0A3/TAUB2I3/ IISA4SDI/IISA4SDO/CSIH2RYI/ CSIH2RYO
TAUB2O4	TBD	TBD	107			P26_4/KR0I4/MEMC0A4/TAUB2I4/ IISA5SDO/CSIH2SC
TAUB2O5	TBD	TBD	72			P3_3/TAUB2I5/IISAACK/KR0I3/ CSIH1CSS3/ETH0REFCLK/ ETH0TXER
	TBD	TBD	108			P26_5/KR0I5/MEMC0A5/TAUB2I5/ IISA5SCK/CSIH2CSS0
TAUB2O6	TBD	TBD	73			P3_4/TAUB2I6/INTP15/IISA3SDO/ KR0I4/CSIH1CSS4/ETH0COL
	TBD	TBD	109			P26_6/KR0I6/MEMC0A6/TAUB2I6/ IISA5WS/CSIH2CSS1
TAUB2O7	TBD	TBD	74			P3_5/TAUB2I7/IISA3SCK/KR0I5/ CSIH1CSS5/ETH0TXCLK
	TBD	TBD	110			P26_7/KR0I7/MEMC0A7/TAUB2I7/ IISA5SDI/IISA5SDO/CSIH2CSS2
TAUB2O8	TBD	TBD	111			P26_8/INTP8/MEMC0A8/TAUB2I8/ IISA3SCK/CSIH2CSS3
TAUB2O9	TBD	TBD	75			P3_6/TAUB2I9/IISA3WS/KR0I6/ CSIH1CSS6/ETH0RXDV
	TBD	TBD	114			P26_9/INTP9/MEMC0A9/TAUB2I9/ IISA3WS/CSIH2CSS4
TAUB2O10	TBD	TBD	76			P3_7/TAUB2I10/IISA3SDI/IISA3SDO/ KR0I7/CSIH1CSS7/ETH0RXCLK
	TBD	TBD	115			P26_10/INTP10/MEMC0A10/ TAUB2I10/IISA2SCK/CSIH2CSS5
TAUB2O11	TBD	TBD	116			P26_11/INTP11/MEMC0A11/ TAUB2I11/IISA2WS/CSIH2CSS6
TAUB2O12	TBD	TBD	117			P26_12/INTP12/MEMC0A12/ TAUB2I12/IISA0SCK/CSIH2CSS7
TAUB2O13	TBD	TBD	118			P26_13/INTP13/MEMC0A13/ TAUB2I13/IISA0WS

Table 2-42 List of pin functions in alphabetical order (26/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
TAUB2O14	TBD	TBD	119	Output	TAUB2 timer output	P26_14/INTP14/MEMC0A14/ TAUB2I14/IISA5SDI/IISA5SDO
TAUB2O15	TBD	TBD	120			P26_15/INTP15/MEMC0A15/ TAUB2I15/IISA4SDI/IISA4SDO
TAUJ0I0	TBD	TBD	22	Input	TAUJ0 timer input	JP0_0/INTP0/TAUJ0O0/DCUTDI
	TBD	TBD	37			P0_0/TAUJ0O0/INTP4/$\overline{\text{CSIG0SSI}}$/ IICB0SDA/$\overline{\text{RESETOUT}}$
TAUJ0I1	TBD	TBD	23			JP0_1/INTP1/TAUJ0O1/DCUTDO
	TBD	TBD	38			P0_1/TAUJ0O1/INTP5/CSIG0SO/ IICB0SCL/FLMD1
TAUJ0I2	TBD	TBD	24			JP0_2/INTP2/TAUJ0O2/DCUTCK
	TBD	TBD	39			P0_2/TAUJ0O2/INTP6/CSIG0SI/ IICB1SDA
TAUJ0I3	TBD	TBD	25			JP0_3/INTP3/TAUJ0O3/DCUTMS
	TBD	TBD	40			P0_3/TAUJ0O3/INTP7/CSIG0SC/ IICB1SCL
TAUJ0O0	TBD	TBD	22	Output	TAUJ0 timer output	JP0_0/INTP0/TAUJ0I0/DCUTDI
	TBD	TBD	37			P0_0/TAUJ0I0/INTP4/$\overline{\text{CSIG0SSI}}$/ IICB0SDA/$\overline{\text{RESETOUT}}$
TAUJ0O1	TBD	TBD	23			JP0_1/INTP1/TAUJ0I1/DCUTDO
	TBD	TBD	38			P0_1/TAUJ0I1/INTP5/CSIG0SO/ IICB0SCL/FLMD1
TAUJ0O2	TBD	TBD	24			JP0_2/INTP2/TAUJ0I2/DCUTCK
	TBD	TBD	39			P0_2/TAUJ0I2/INTP6/CSIG0SI/ IICB1SDA
TAUJ0O3	TBD	TBD	25			JP0_3/INTP3/TAUJ0I3/DCUTMS
	TBD	TBD	40			P0_3/TAUJ0I3/INTP7/CSIG0SC/ IICB1SCL
URTE0RX	TBD	TBD	42	Input	URTE0 serial receive data input	P0_5/FCN0RX/$\overline{\text{IEBB0TX}}$/INTP9/ CSIH1SO/IICB2SCL
	TBD	TBD	145			P21_14/$\overline{\text{MEMC0HLDAK}}$/FCN0TX/ CSIG4SI/IICB2SCL
URTE0TX	TBD	TBD	41	Output	URTE0 serial transmit data output	P0_4/$\overline{\text{IEBB0RX}}$/FCN0TX/INTP8/ CSIH1SI/IICB2SDA
	TBD	TBD	146			P21_15/$\overline{\text{MEMC0ASTB}}$/CSIG4SC
URTE1RX	TBD	TBD	49	Input	URTE1 serial receive data input	P1_0/TAUA0I0/TAUA0O0/IISAACK/ ETH0RXER/CSIH0CSS0
	TBD	TBD	138			P21_7/$\overline{\text{MEMC0WAIT}}$/CSIG0SC/ IICB1SCL
URTE1TX	TBD	TBD	50	Output	URTE1 serial transmit data output	P1_1/TAUA0I1/TAUA0O1/IISA0SCK/ $\overline{\text{CSIG4SSI}}$/ETH0RXD0/CSIH0CSS1
	TBD	TBD	137			P21_6/MEMC0CLK/CSIG0SI/ IICB1SDA

Table 2-42 List of pin functions in alphabetical order (27/27)

Pin name	Pin number			I/O	Function	Alternate-function pin
	SG4-H	SJ4-H	SK4-H			
URTE2RX	TBD	TBD	61	Input	URTE2 serial receive data input	P1_10/TAUA0I10/TAUA0O10/IISA2WS/ENCA1AIN/CSIH0CSS5/ETH0MDC
	TBD	TBD	134			P21_5/$\overline{\text{MEMC0RD}}$/$\overline{\text{IEBB0TX}}$/CSIG0SO/IICB0SCL
URTE2TX	TBD	TBD	62	Output	URTE2 serial transmit data output	P1_11/TAUA0I11/TAUA0O11/IISA2SDI/IISA2SDO/ENCA1BIN/CSIH0CSS6/ETH0CRSDV
	TBD	TBD	133			P21_4/$\overline{\text{MEMC0WR}}$/$\overline{\text{IEBB0RX}}$/$\overline{\text{CSIG0SSI}}$/$\overline{\text{MEMC0WE}}$/IICB0SDA
URTE3RX	TBD	TBD	66	Input	URTE3 serial receive data input	P1_15/TAUA0I15/TAUA0O15/PCM0SI/PCM0SO/IICB3SCL
URTE3TX	TBD	TBD	65	Output	URTE3 serial transmit data output	P1_14/TAUA0I14/TAUA0O14/PCM0SEN/ENCA1TIN1/IICB3SDA
URTE10RX	TBD	TBD	44	Input	URTE10 serial receive data input	P0_7/FCN1RX/INTP11/CSIH1SC
	TBD	TBD	128			P27_5/INTP5/MEMC0A21/IISA0WS
URTE10TX	TBD	TBD	43	Output	URTE10 serial transmit data output	P0_6/FCN1TX/INTP10/CSIH1RYI/CSIH1RYO/$\overline{\text{CSIH1SSI}}$
	TBD	TBD	127			P27_4/INTP4/MEMC0A20/IISA0SCK
<R> WAKE	TBD	TBD	46	Output	DEEPSTOP mode indicator	–
X1	TBD	TBD	28	Input	Connection of resonator for main clock	–
X2	TBD	TBD	27	–		–
XT1	TBD	TBD	31	Input	Connection of resonator for subclock	–
XT2	TBD	TBD	32	–		–

2.4.7 Port statuses during and after reset and after entering or exiting standby mode

TBD

2.4.8 Recommended connection of unused pins

Output ports and I/O ports become high impedance after reset is cancelled, and their input buffers are unable to be used. (PODCn_m = 1, PIBCn_m = 0).

Unused pins can therefore be left unconnected.

This rule does not apply to the following pins:

- JP0_4: The internal pull-down resistor is active
- P0_0: The RESETOUT signal is output
- $\overline{\text{RESET}}$: Input must be maintained
- WAKE: Always output
- IC: Be sure to input a low-level voltage

<R>

We recommend connecting unused pins as follows:

Pin	Recommended connection
JTAG port 0 (other than JP0_4) and port 0 (other than P0_0 to P0_3)	Output: Leave open. Input: Independently connect to E0VDD or E0VSS via a resistor.
JP0_4 P0_0 to P0_3	Output: Leave open. Input: Independently connect to E0VSS via a resistor.
Ports 1 to 3	Output: Leave open. Input: Independently connect to E1VDD or E1VSS via a resistor.
Port 10	Output: Leave open. Input: Independently connect to A0VDD or A10VSS via a resistor.
Ports 21, and 24 to 28	Output: Leave open. Input: Independently connect to B0VDD or B0VSS via a resistor.
PTCTL1	Leave open.
FLMD0	Connect to E0VSS.
$\overline{\text{RESET}}$	Connect to E0VDD.
WAKE	Leave open.
XT1	Connect to OSCVSS.
XT2	Leave open.
MLBA0SIG, MEBA0DAT, MLBA0CLK	Independently connect to MLB0VSS via a resistor.
A0VDD, A0VREFP	Connect to E1VDD.
A0VSS, A0VREFM	Connect to E1VSS.

2.5 Port Filters

The input signals at some pins are passed through a filter to remove noise and glitches. The V850E2/Sx4-H supports both analog and digital filters.

The first part of this section provides an overview of the port filters, and describes which port input signals are equipped with which kind of filter, their control registers and bits, and the register addresses.

Subsequent sections provide a detailed description of the analog and digital filter features and their control registers.

For details, see *2.6 "Description of Port Filters"*.

2.5.1 Port filter assignment

A list of the input pins that incorporate an analog or digital filter is provided below.

(1) Input pins that incorporate analog filter type A

Analog filter type A is controlled by the following register:

- Filter control register FCLAnCTLm (m = 0 to 7)
  A dedicated FCLAnCTLm register is provided for each pin in a port that incorporates an analog filter. "n" refers to the port group, and "m" refers to one of eight signals input to the pins in that port group.

Table 2-43 Input signals and control registers for ports that incorporate analog filter type A (1/2)

Input signal	Control register		
	Register		Address
INTP0	FCLA0	CTL0	FF41 4000$_H$
INTP1		CTL1	FF41 4004$_H$
INTP2		CTL2	FF41 4008$_H$
INTP3		CTL3	FF41 400C$_H$
INTP4		CTL4	FF41 4010$_H$
INTP5		CTL5	FF41 4014$_H$
INTP6		CTL6	FF41 4018$_H$
INTP7		CTL7	FF41 401C$_H$
INTP8	FCLA1	CTL0	FF41 4020$_H$
INTP9		CTL1	FF41 4024$_H$
INTP10		CTL2	FF41 4028$_H$
INTP11		CTL3	FF41 402C$_H$
INTP12		CTL4	FF41 4030$_H$
INTP13		CTL5	FF41 4034$_H$
INTP14		CTL6	FF41 4038$_H$
INTP15		CTL7	FF41 403C$_H$
NMI	FCLA2	CTL0	FF41 4040$_H$

Table 2-43 Input signals and control registers for ports that incorporate analog filter type A (2/2)

Input signal	Control register		
	Register		Address
KR0I0	FCLA3	CTL0	FF41 4060$_H$
KR0I1		CTL1	FF41 4064$_H$
KR0I2		CTL2	FF41 4068$_H$
KR0I3		CTL3	FF41 406C$_H$
KR0I4		CTL4	FF41 4070$_H$
KR0I5		CTL5	FF41 4074$_H$
KR0I6		CTL6	FF41 4078$_H$
KR0I7		CTL7	FF41 407C$_H$

(2) Input pins that incorporate analog filter type B

Analog filter type B is controlled by the following register:

- Filter control register FCLAnCTLm (m = 0 to 7)

  A dedicated FCLAnCTLm register is provided for each pin in a port that incorporates an analog filter. "n" refers to the port group, and "m" refers to one of eight signals input to the pins in that port group.

Table 2-44 Input signals and control registers for ports that incorporate analog filter type B

Input signal	Control register		
	Register		Address
TAUJ0I0	FCLA4	CTL0	FF41 4080$_H$
TAUJ0I1		CTL1	FF41 4084$_H$
TAUJ0I2		CTL2	FF41 4088$_H$
TAUJ0I3		CTL3	FF41 408C$_H$

(3) Input pins that incorporate analog filter type C

There is no register that controls analog filter type C.

Table 2-45 Input signals and control registers for ports that incorporate analog filter type C

Input signal	Control register		
	Register		Address
Always-On area:			
FLMD0	–	–	–
FLMD1		–	–
$\overline{\text{RESET}}$		–	–

(4) Input pins that incorporate digital filter type D

Digital filter type D is controlled by the following registers:

- Filter control register FCLAnCTLm (m = 0 to 7)

  A dedicated FCLAnCTLm register is provided for each pin in a port that incorporates a digital filter. "n" refers to the port group, and "m" refers to one of eight signals input to the pins in that port group.

- Digital noise elimination control register DNFAnCTL

  Each DNFAnCTL register controls the digital filter processing for port group n. (Note that each port group can include up to 16 input signals.)

- Digital noise elimination enable register DNFAnEN

  The digital filter processing for port group n (each of which can include up to 16 input signals) is enabled and disabled by setting the DNFAnEN bit of the DNFAnFEN[15:0] register.

Table 2-46 Input signals and control registers for ports that incorporate digital filter type D (1/3)

Input signal	Control register						
	Register	Address	Filter enable bit		Register		Address
TAUA0I0	DNFA0CTL DNFA0EN DNFA0ENH DNFA0ENL	FF41 1000$_H$ FF41 1004$_H$ FF41 1008$_H$ FF41 100C$_H$	DNFA0EN.DNFA0	NFEN0	FCLA5	CTL0	FF41 5000$_H$
TAUA0I1				NFEN1		CTL1	FF41 5004$_H$
TAUA0I2				NFEN2		CTL2	FF41 5008$_H$
TAUA0I3				NFEN3		CTL3	FF41 500C$_H$
TAUA0I4				NFEN4		CTL4	FF41 5010$_H$
TAUA0I5				NFEN5		CTL5	FF41 5014$_H$
TAUA0I6				NFEN6		CTL6	FF41 5018$_H$
TAUA0I7				NFEN7		CTL7	FF41 501C$_H$
TAUA0I8				NFEN8	FCLA6	CTL0	FF41 5020$_H$
TAUA0I9				NFEN9		CTL1	FF41 5024$_H$
TAUA0I10				NFEN10		CTL2	FF41 5028$_H$
TAUA0I11				NFEN11		CTL3	FF41 502C$_H$
TAUA0I12				NFEN12		CTL4	FF41 5030$_H$
TAUA0I13				NFEN13		CTL5	FF41 5034$_H$
TAUA0I14				NFEN14		CTL6	FF41 5038$_H$
TAUA0I15				NFEN15		CTL7	FF41 503C$_H$

Table 2-46 Input signals and control registers for ports that incorporate digital filter type D (2/3)

Input signal	Control register						
	Register	Address	Filter enable bit		Register		Address
URTE10RX	DNFA1CTL DNFA1EN DNFA1ENH DNFA1ENL	FF41 1020$_H$ FF41 1024$_H$ FF41 1028$_H$ FF41 102C$_H$	DNFA1EN.DNFA1	NFEN0	FCLA7	CTL0	FF41 5040$_H$
CSIG4SC				NFEN2		CTL2	FF41 5048$_H$
CSIG4SI				NFEN3		CTL3	FF41 504C$_H$
$\overline{\text{CSIG4SSI}}$				NFEN5		CTL5	FF41 5054$_H$
ADCA0TRG0				NFEN8	FCLA8	CTL0	FF41 5060$_H$
ADCA0TRG1				NFEN9		CTL1	FF41 5064$_H$
ADCA0TRG2				NFEN10		CTL2	FF41 5068$_H$
ENCA0AIN				NFEN11		CTL3	FF41 506C$_H$
ENCA0BIN				NFEN12		CTL4	FF41 5070$_H$
ENCA0ZIN				NFEN13		CTL5	FF41 5074$_H$
ENCA0TIN0				NFEN14		CTL6	FF41 5078$_H$
ENCA0TIN1				NFEN15		CTL7	FF41 507C$_H$
TAUB2I0	DNFA5CTL DNFA5EN DNFA5ENH DNFA5ENL	FF41 2060$_H$ FF41 2064$_H$ FF41 2068$_H$ FF41 206C$_H$	DNFA5EN.DNFA5	NFEN0	FCLA15	CTL0	FF41 60C0$_H$
TAUB2I1				NFEN1		CTL1	FF41 60C4$_H$
TAUB2I2				NFEN2		CTL2	FF41 60C8$_H$
TAUB2I3				NFEN3		CTL3	FF41 60CC$_H$
TAUB2I4				NFEN4		CTL4	FF41 60D0$_H$
TAUB2I5				NFEN5		CTL5	FF41 60D4$_H$
TAUB2I6				NFEN6		CTL6	FF41 60D8$_H$
TAUB2I7				NFEN7		CTL7	FF41 60DC$_H$
TAUB2I8				NFEN8	FCLA16	CTL0	FF41 60E0$_H$
TAUB2I9				NFEN9		CTL1	FF41 60E4$_H$
TAUB2I10				NFEN10		CTL2	FF41 60E8$_H$
TAUB2I11				NFEN11		CTL3	FF41 60EC$_H$
TAUB2I12				NFEN12		CTL4	FF41 60F0$_H$
TAUB2I13				NFEN13		CTL5	FF41 60F4$_H$
TAUB2I14				NFEN14		CTL6	FF41 60F8$_H$
TAUB2I15				NFEN15		CTL7	FF41 60FC$_H$
ENCA1AIN	DNFA8CTL DNFA8EN DNFA8ENH DNFA8ENL	FF41 20C0$_H$ FF41 20C4$_H$ FF41 20C8$_H$ FF41 20CC$_H$	DNFA8EN.DNFA8	NFEN8	FCLA30	CTL0	FF41 62A0$_H$
ENCA1BIN				NFEN9		CTL1	FF41 62A4$_H$
ENCA1ZIN				NFEN10		CTL2	FF41 62A8$_H$
ENCA1TIN0				NFEN11		CTL3	FF41 62AC$_H$
ENCA1TIN1				NFEN12		CTL4	FF41 62B0$_H$

Table 2-46 Input signals and control registers for ports that incorporate digital filter type D (3/3)

Input signal	Control register						
	Register	Address	Filter enable bit		Register		Address
CSIH0SC	DNFA9CTL DNFA9EN DNFA9ENH DNFA9ENL	FF41 20E0$_H$ FF41 20E4$_H$ FF41 20E8$_H$ FF41 20EC$_H$	DNFA9EN.DNFA9	NFEN0	FCLA22	CTL0	FF41 61A0$_H$
CSIH0RYI				NFEN1		CTL1	FF41 61A4$_H$
CSIH0SI				NFEN2		CTL2	FF41 61A8$_H$
$\overline{\text{CSIH0SSI}}$				NFEN3		CTL3	FF41 61AC$_H$
CSIH1SC				NFEN4		CTL4	FF41 61B0$_H$
CSIH1RYI				NFEN5		CTL5	FF41 61B4$_H$
CSIH1SI				NFEN6		CTL6	FF41 61B8$_H$
$\overline{\text{CSIH1SSI}}$				NFEN7		CTL7	FF41 61BC$_H$
CSIH2SC				NFEN8	FCLA23	CTL0	FF41 61C0$_H$
CSIH2RYI				NFEN9		CTL1	FF41 61C4$_H$
CSIH2SI				NFEN10		CTL2	FF41 61C8$_H$
$\overline{\text{CSIH2SSI}}$				NFEN11		CTL3	FF41 61CC$_H$
CSIG0SC	DNFA10CTL DNFA10EN DNFA10ENH DNFA10ENL	FF41 2100$_H$ FF41 2104$_H$ FF41 2108$_H$ FF41 210C$_H$	DNFA10EN.DNFA10	NFEN0	FCLA24	CTL0	FF41 61E0$_H$
CSIG0SI				NFEN2		CTL2	FF41 61E8$_H$
$\overline{\text{CSIG0SSI}}$				NFEN3		CTL3	FF41 61EC$_H$
URTE0RX	DNFA11CTL DNFA11EN DNFA11ENH DNFA11ENL	FF41 2120$_H$ FF41 2124$_H$ FF41 2128$_H$ FF41 212C$_H$	DNFA11EN.DNFA11	NFEN4	FCLA26	CTL4	FF41 6230$_H$
URTE1RX				NFEN5		CTL5	FF41 6234$_H$
<R> URTE2RX				NFEN8	FCLA27	CTL0	FF41 6240$_H$
<R> URTE3RX				NFEN9		CTL1	FF41 6244$_H$

<R> **Cautions**

1. Port filters are assigned to the input pins of the clocked serial interface modules (CSIGn and CSIHn), and these filters are enabled as the initial setting. However, because the use of the port filters might cause a communication error, do not use the port filters when using CSIGn or CSIHn; instead, enable the filter bypass by setting the corresponding registers as shown below.

   CSIG0SC: FCLA24CTL0 = 80H, CSIG0SI: FCLA24CTL2 = 80H, $\overline{\text{CSIG0SSI}}$: FCLA24CTL3 = 80H,

   CSIG4SC: FCLA7CTL2 = 80H, CSIG4SI: FCLA7CTL3 = 80H, $\overline{\text{CSIG4SSI}}$: FCLA7CTL5 = 80H,

   CSIH0SC: FCLA22CTL0 = 80H, CSIH0RYI: FCLA22CTL1 = 80H, CSIH0SI: FCLA22CTL2 = 80H, $\overline{\text{CSIH0SSI}}$: FCLA22CTL3 = 80H,

   CSIH1SC: FCLA22CTL4 = 80H, CSIH1RYI: FCLA22CTL5 = 80H, CSIH1SI: FCLA22CTL6 = 80H, $\overline{\text{CSIH1SSI}}$: FCLA22CTL7 = 80H,

   CSIH2SC: FCLA23CTL0 = 80H, CSIH2RYI: FCLA23CTL1 = 80H, CSIH2SI: FCLA23CTL2 = 80H, $\overline{\text{CSIH2SSI}}$: FCLA23CTL3 = 80H

<R> 2. A port filter is assigned to the receive data input pin (URTEnRX) of asynchronous serial interface E (UARTEn), and this filter is enabled as the initial setting. However, because UARTEn has an internal filter, do not use the port filter when using UARTEn; instead, enable the filter bypass by setting the corresponding registers as shown below.

URTE0RX: FCLA26CTL4 = 80H, URTE1RX: FCLA26CTL5 = 80H, URTE2RX: FCLA27CTL0 = 80H, URTE3RX: FCLA27CTL1 = 80H, URTE10RX: FCLA7CTL0 = 80H

2.5.2 Port filter clock supply

The clock supply for each filter type in each port domain is shown in the table below.

Table 2-47 Port filter clock supply

Port domain	Filter type	Filter clock	Connected to:
Always-On area	Analog type A	PCLK	Clock controller CKSCLK_A02
	Analog type B	PCLK	Clock controller CKSCLK_A02
	Analog type C	–	–
Isolated area 0	Digital type D	PCLK	Clock controller CKSCLK_005
		DNFATCKI	Clock controller CKSCLK_016
Isolated area 1	Digital type D	PCLK	Clock controller CKSCLK_101
		DNFATCKI	Clock controller CKSCLK_128

2.6 Description of Port Filters

External input signals pass through different types of filters according to the application of the signal to be filtered.

- Analog filters

  Analog filters have fixed characteristics.

  – Type A: This filter type is used to pass through signals used for edge or level detection.

    The signals output from this filter are used to transmit an external event. The filter does not preserve the timing of the external signal, but it does preserve the level or a change in the level of the signal.

    An external interrupt is a typical example of an event signal.

  – Type B: This filter type is used to pass through signals used for the filter bypass option.

    This filter preserves the timing of the input signal when it is output from the filter. This filter can also be bypassed.

    A timer input signal used to measure the frequency is a typical example of a signal input to this filter.

  – Type C: A simple analog filter

    The input signals always pass through an analog filter. This filter cannot be bypassed.

    This type of filter is usually used for inputting the external $\overline{\text{RESET}}$signal or mode signals.

- Digital filters

  The digital filter characteristics can be adjusted to suit the application.

  – Type D: This filter type can be specified to pass through signals used for the filter bypass option.

    This filter preserves the timing of the input signal when it is output from the filter. This filter can also be bypassed.

    A timer input signal used to measure the frequency is a typical example of a signal input to this filter.

2.6.1 Analog filters

Analog filter characteristics

The characteristics of the analog filters, as well as of the level and edge detectors, are specified in the target electrical specifications.

Analog filter control registers

A dedicated control register FCLAnCTLm is provided for each input pin in a port that incorporates an analog filter.

The FCLAnCTLm registers are ordered in groups of 8 registers with the index n. The register index m is a number from 0 to 7.

FCLA group n: FCLAnCTL0 to FCLAnCTL7

The correspondance between input signals and control registers and the address of each register are shown in the tables in *2.5.1 “Port filter assignment”*.

The status of analog filters in standby modes

- DEEPSTOP mode

  The analog filters that are assigned to the power domains of Isolated area 0 and Isolated area 1 stop operating in DEEPSTOP mode because these power domains stop supplying power in DEEPSTOP mode.
  Analog filters assigned to the Always-On area (AWO) continue to operate.

- STOP mode

  The operation of the analog filter and how the filter is woken up depend on the filter type. See the description of analog filter types below for details.

(1) Analog filter type A

A block diagram of analog filter type A is shown below.

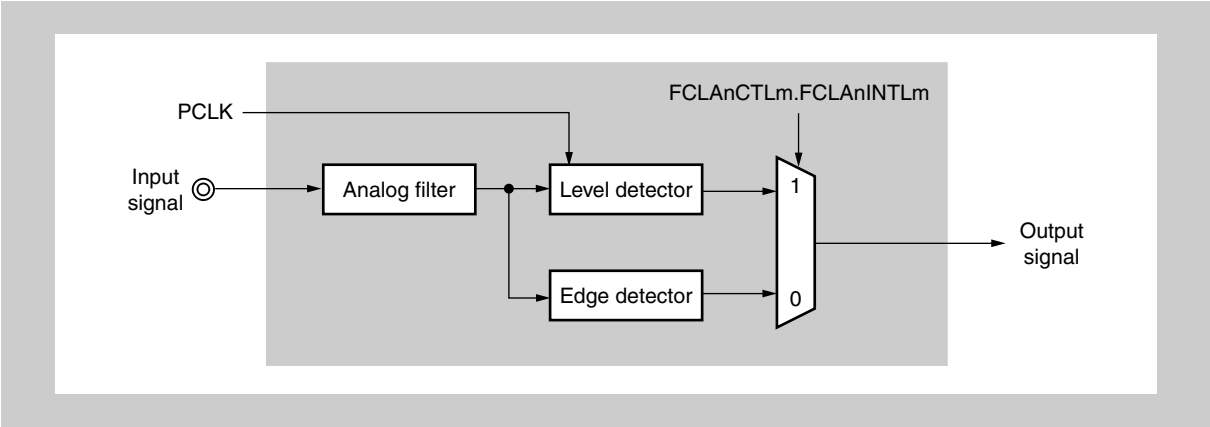


Figure 2-7 Block diagram of analog filter type A

After passing an external signal through the filter to eliminate noise and spikes, the filter generates an output signal according to whether an event is detected; that is whether a specified level is detected or whether a change in the level (an edge) occurs.

Whether a level or an edge is detected is selected by the control bit FCLAnCTLm.FCLAnINTLm.

- FCLAnINTLm = 0: Edge detection

  Whether a rising or falling edge is detected can be specified by setting the FCLAnCTLm.FCLAnINTRm and FCLAnCTLm.FCLAnINTFm bits.

- FCLAnINTLm = 1: Level detection

  The detection of a high level or low level can be specified by setting FCLAnCTLm.FCLAnINTRm.

The table below summarizes the detection conditions of the analog filter.

Table 2-48 Analog filter event detection conditions

FCLAnINTLm	FCLAnINTFm	FCLAnINTRm	Edge detection	Level detection
0	0	0	No edge detected	Disabled
	0	1	Rising edge	
	1	0	Falling edge	
	1	1	Rising and falling edges	
1	X	0	Disabled	Low level
	X	1		High level

Status of analog filter type A in STOP mode

If the PCLK clock is stopped in STOP mode, only analog filter type A can execute edge detection. Therefore, if the input signal is to be used as the STOP mode wakeup signal, select the edge to be detected by setting the FCLAnINTLm bit to 0.

(2) Analog filter type B

A block diagram of analog filter type B is shown below.

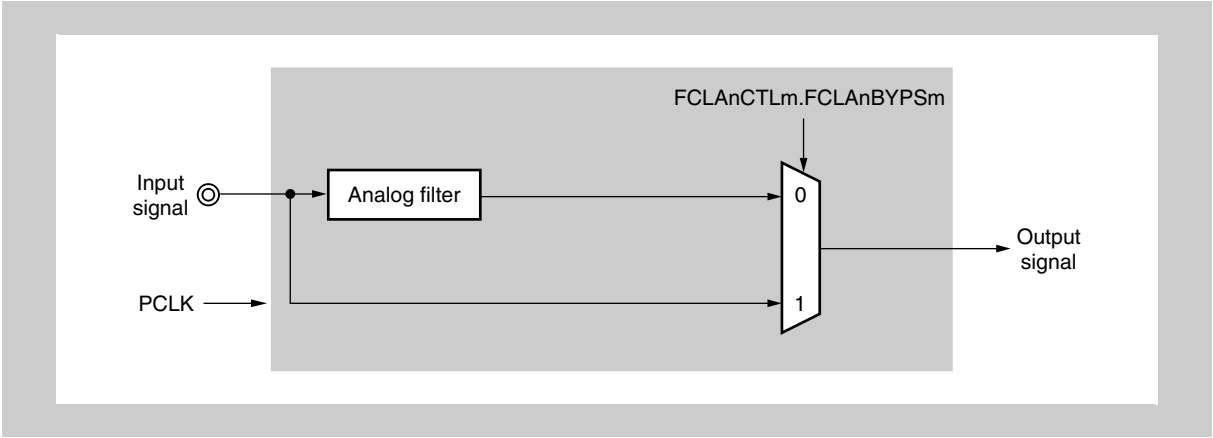


Figure 2-8 Block Diagram of analog filter type B

This filter can be specified to be bypassed.

- DCLAnCTLm.FCLAnBYPSm = 0: The signal that passed through the filter is output.
- DCLAnCTLm.FCLAnBYPSm = 1: The signal that has not passed through the filter is output.

Status of analog filter type B in STOP mode

The signal output from analog filter type B can always be used as the STOP mode wakeup signal.

(3) Analog filter type C

A block diagram of analog filter type C is shown below.

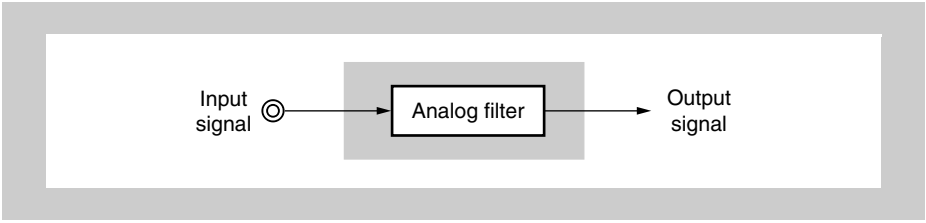


Figure 2-9 Block Diagram of analog filter type C

The output signals are always signals that have passed through an analog filter.

Status of analog filter type C in STOP mode

The signal output from analog filter C can always be used as the STOP mode wakeup signal.

2.6.2 Digital filters

Digital filter characteristics

The digital filter characteristics can be adjusted to suit the application.

The input signal is sampled at a sampling frequency of f_s.

If the specified number of successive samples yields the same (high or low) level, the signal level is judged as valid and the filter output signal is set accordingly.

If the external signal level changes within the specified number of samples (same level samples), the signal level is judged as noise or a spike. In this case, the filter output signal does not change.

The pulse length used to judge whether an input signal is a noise depends on the sampling frequency and the specified number of samples.

Either of the following parameters can be specified:

- By setting the DNFAnCTL.DNFAnPRS[1:0] bits, the sampling frequency can be selected based on $f_s = f_{DNFATCKI} / 2^{DNFAnPRS[1:0]}$, where $f_{DNFATCKI}$ is the frequency of the DNFATCKI clock.
- Use DNFAnCTL.DNFAnNFSTS[1:0] to specify a number of samples from 2 to 5.

External signal pulses shorter than

(Number of samples – 1) x $1/f_s$

are always suppressed.

External signal pulses longer than

(Number of samples) x $1/f_s$

are always judged as valid and are passed on to the filter output.

External signal pulses with a width between

(Number of samples – 1) x $1/f_s$ and (Number of samples) x $1/f_s$

may be suppressed or judged as valid.

The filter operation is illustrated in the figure below with DNFAnNFSTS[1:0] = 01_B (3 same-level samples).

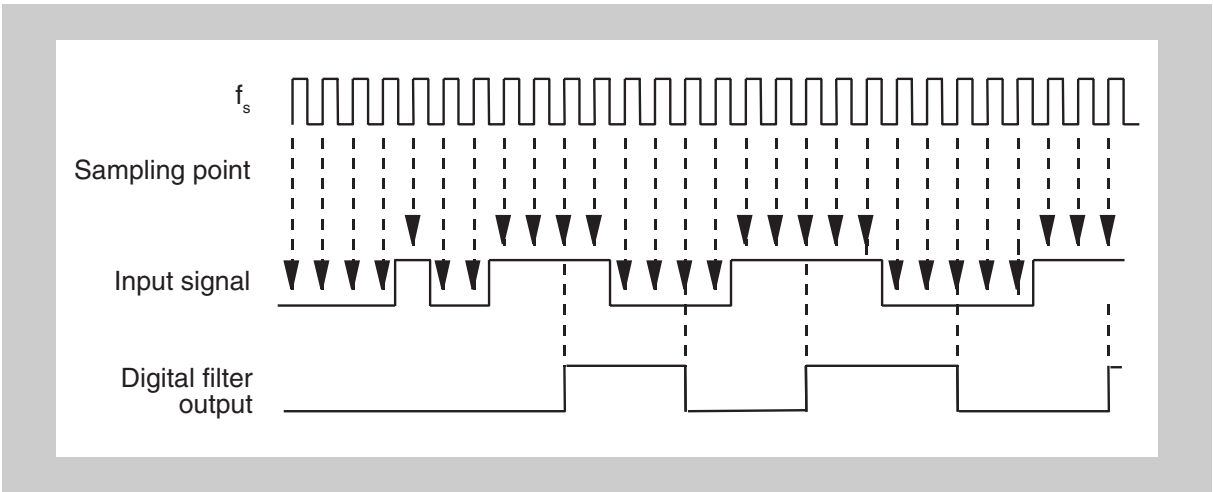


Figure 2-10 Digital filter operation

Digital filter groups

The signals that are input to pins that incorporate digital filters are ordered in groups of up to 16 signals.

The digital filter characteristics specified by DNFAnCTL.DNFAnPRS[1:0] and DNFAnNFSTS[1:0] apply to the signals of the entire group.

However the digital filter for each signal can be enabled and disabled separately by using DNFAnEN.DNAFnNFENm.

Cautions

1. When inputting the digital filter output signal to an alternate-function pin, enable the digital filter (DNFAnEN.DNAFnENm = 1), wait for the time shown below, and then set the port to its alternative mode.

   DNFAnNFSTS[1:0] x $1/f_s$ + 4 x $1/f_{DNFATCKI}$

<R> 2. When using an event signal output from the digital filter as an interrupt source, enable the digital filter (DNFAnEN.DNAFnENm = 1) while interrupts are disabled. Then wait for the time shown below, clear the interrupt request flag, and then enable interrupts.

   DNFAnNFSTS[1:0] x $1/f_s$ + 5 x $1/f_{DNFATCKI}$

Status of digital filters in standby modes

- DEEPSTOP mode

  The digital filters are assigned to the power domains of Isolated area 0 and Isolated area 1, which stop supplying power in DEEPSTOP mode. The digital filters therefore stop operating in DEEPSTOP mode.

- STOP mode

  To perform digital noise elimination in STOP mode, the DNFATCKI clock must be supplied. Therefore, if DNFATCKI is stopped in STOP mode, signals that are passed through a digital filter cannot be used as STOP mode wakeup events.

  If DNFATCKI is being supplied in STOP mode, an external signal can be used as a wakeup event.

Digital filter control registers

A common digital filter setup register DNFAnCTL and digital filter enable register DNFAnEN is provided for each group of up to 16 digital filters (indexed by the group number n). The settings of these registers apply to all the filters in a group.

While the filter setting specified by DNFAnCTL applies to the whole group, each filter can be enabled and disabled separately by using the DNFAnENm control bit in the filter enable register DNFAnEN. The register index m is a number from 0 to 15.

> The DNFAnCTL register controls group n consisting of the digital filters indexed by m (m = 0 to 15). Each filter can be enabled and disabled by using the control bits DNFAnEN.DNAFnEN0 to DNFAnEN.DNAFnEN15.

Edge detection settings are specified by using the dedicated filter control register FCLAnCTLm.

The FCLAnCTLm registers are ordered in groups of 8 registers with the index n. The register index m is a number from 0 to 7.

> FCLA group n: FCLAnCTL0 to FCLAnCTL7

The allocation of input signals to control registers and the address of each register are shown in *Table 2-46 "Input signals and control registers for ports that incorporate digital filter type D"* in *2.5.1 "Port filter assignment"*.

Caution Changing any control register settings while the corresponding digital filter is enabled (which is specified by setting the DNFAnEN.DNAFnENm bit to 1) might result in an unexpected output.

(1) Digital filter type D

A block diagram of digital filter type D is shown below.

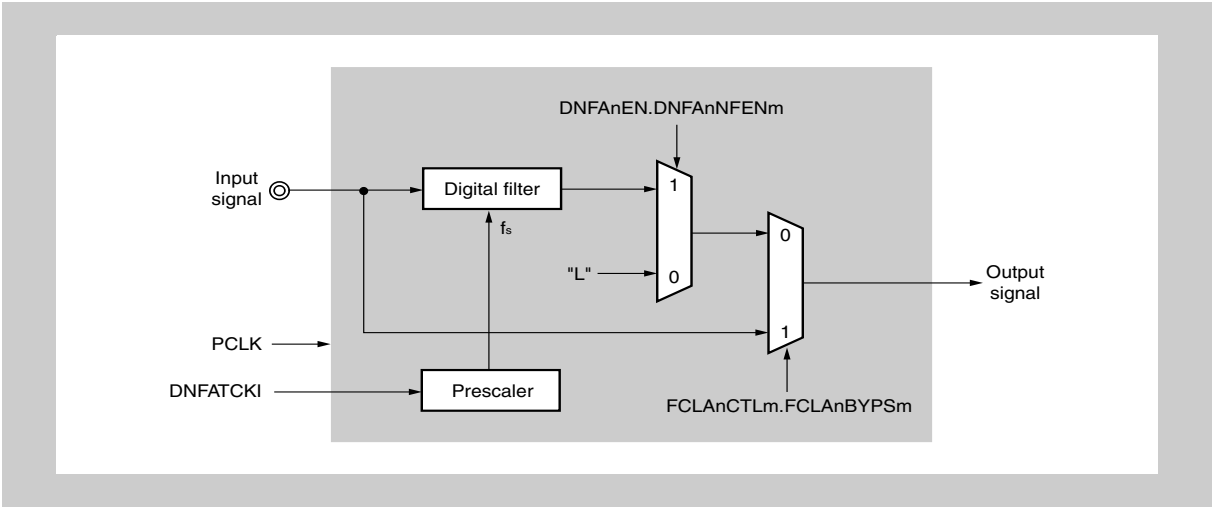


Figure 2-11 Block diagram of digital filter type D

The output signal depends on the register setting, as shown in the following table.

Table 2-49 Output options for digital filter type D

FCLAnCTLm.FCLAnBYPSm	DNFAnEN.DNFAnNFENm	Output signal
0	0	Fixed to low level
	1	Input signal passed through filter
1	X	Input signal not passed through filter

2.6.3 Filter control registers

The analog and digital filters are controlled and operated by the following registers:

Table 2-50 List of filter registers

Register name	Symbol	Address
Filter control register m	FCLAnCTLm	The addresses are shown in the tables in *2.5.1 "Port filter assignment"*.
Digital noise elimination control register	DNFAnCTL	
Digital noise elimination enable register	DNFAnEN	

(1) FCLAnCTLm – Filter control register

This register controls the analog and digital filter operation.

Because the control options for analog and digital filters partially differ, register descriptions are provided separately.

Access This register can be read or written in 8-bit or 1-bit units.

Address The allocation of input signals to FCLAnCTLm registers and the address of each register are shown in the tables in *2.5.1 "Port filter assignment"*.

Initial value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
FCLAn BYPSm	0	0	0	0	FCLAn INTLm	FCLAn INTFm	FCLAn INTRm
R/W	R	R	R	R	R/W	R/W	R/W

Table 2-51 FCLAnCTLm register contents

Bit position	Bit name	Function
7	FCLAnBYPSm	Filter bypass control: 0: Filter bypass disabled 1: Filter bypass enabled **Note** This bit is only valid for analog filter type B and digital filter type D.
2	FCLAnINTLm	Detection mode selection 0: Edge detection 1: Level detection **Note** This bit is only valid for analog filter type A.
1	FCLAnINTFm	• In level detection mode (FCLAnINTLm = 1): This bit has no effect. 0: Low level detection 1: High level detection • In edge detection mode (FCLAnINTLm = 0): Falling edge detection control 0: Falling edge detection disabled 1: Falling edge detection enabled **Note** This bit is only valid for analog filter type A.
0	FCLAnINTRm	• In level detection mode (FCLAnINTLm = 1): Detected level selection 0: Low level detection 1: High level detection • In edge detection mode (FCLAnINTLm = 0): Rising edge detection control 0: Rising edge detection disabled 1: Rising edge detection enabled **Note** This bit is only valid for analog filter type A.

(2) DNFAnCTL – Digital noise elimination control register

This register is used to specify the filter characteristics of the digital noise elimination filter.

Note This register is only valid for digital filter type D.

Access These registers can be read or written in 8-bit units.

Address The allocation of input signals to the DNFAnCTL registers and the address of each register are shown in *Table 2-46 "Input signals and control registers for ports that incorporate digital filter type D"* in *2.5.1 "Port filter assignment"*.

Initial value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	DNFAnNFSTS[1:0]		0	0	DNFAnPRS[2:0]		
R	R/W	R/W	R	R	R/W	R/W	R/W

Table 2-52 DNFAnCTL register contents

Bit position	Bit name	Function
6, 5	DNFAn NFSTS[1:0]	The DNFAnNFSTS[1:0] bits specify the number of samples used to judge whether an external signal pulse is valid.
2 to 0	DNFAn PRS[2:0]	Digital filter sampling clock selection

DNFAnNFSTS[1:0]	Number of samples
00_B	2
01_B	3
10_B	4
11_B	5

DNFAnPRS[2:0]	Sampling clock frequency
000_B	DNFATCKI/1
001_B	DNFATCKI/2
010_B	DNFATCKI/4
011_B	DNFATCKI/8
100_B	DNFATCKI/16
101_B	DNFATCKI/32
110_B	DNFATCKI/64
111_B	DNFATCKI/128

(3) DNFAnEN – Digital noise elimination enable register

This register enables and disables digital noise elimination for a specified input signal.

Note This register is only valid for digital filter type D.

Access This register can be read or written in 16-bit units.

The higher byte DNFAnNFEN[15:8] register and lower byte DNFAnNFEN[7:0] register can also be separately accessed in 8-bit and 1-bit units by setting the DNFAnENH.DNFAnNFEN[15:8] and DNFAnENL.DNFAnNFEN[7:0] bits.

Address The allocation of input signals to the DNFAnEn registers and the address of each register are shown in *Table 2-46 "Input signals and control registers for ports that incorporate digital filter type D"* in *2.5.1 "Port filter assignment"*.

Initial value 00_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
DNFAn NFEN15	DNFAn NFEN14	DNFAn NFEN13	DNFAn NFEN12	DNFAn NFEN11	DNFAn NFEN10	DNFAn NFEN9	DNFAn NFEN8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
DNFAn NFEN7	DNFAn NFEN6	DNFAn NFEN5	DNFAn NFEN4	DNFAn NFEN3	DNFAn NFEN2	DNFAn NFEN1	DNFAn NFEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-53 DNFAnEN register contents

Bit position	Bit name	Function
15 to 0	DNFAn NFEN[15:0]	Digital noise elimination control 0: Digital noise elimination disabled 1: Digital noise elimination enabled

Chapter 3 CPU System Function

This chapter describes the CPU and its operating modes, address spaces, and memory space registers.

3.1 Overview

The CPU in the V850E2/Sx4-H is based on Harvard architecture and supports the RISC instruction set. The CPU incorporates a 7-stage pipeline that allows almost all instructions to be executed in a single clock cycle, raising the instruction execution speed.

The CPU also includes a 32-bit hardware multiplier for executing multiplication processing, and can execute saturated operation and bit manipulation instructions, making it ideal for use in digital control applications.

CPU

- Core: V850E2M CPU

  Architecture: V850E2v3 architecture
- Instruction execution time

Product	Minimum instruction execution time[a]	Fastest CPU clock[a]
V850E2/Sx4-H	6.25 ns	160 MHz

a) When the SSCG (spread spectrum clock generator) is being used for the CPU clock, the fastest CPU clock might not be 160 MHz because the clock must be modulated. The minimum instruction execution time might therefore differ from this value.

- 32 x 32-bit general-purpose registers
- 7-stage pipeline
- 2-way superscalar
- 32-bit internal architecture
- Memory space:
  - 4 GB linear program space
  - 4 GB linear data space

- Processor protection
  - Memory protection unit (MPU)

    This unit is used to protect the CPU memory space from operations or data manipulations not permitted in the user's program. Up to five instruction and constant protection areas and up to six data protection areas can be set.
  - System register protection (SRP)

    System registers can be protected from damage due to access by an unreliable program.
  - Peripheral protection unit (PPU)

    The peripheral protection unit can detect illegal accesses specified for individual systems, protecting the peripherals from this kind of access.
  - Timing supervision unit (TSU)

    This unit is used to prevent an unreliable program from having an unnecessarily large CPU occupancy rate and to control the system assets and interrupt-disabled time.

Instruction set

- The V850E2M instruction set is backward-compatible with the V850 instruction set but features a reduced code size enabling faster instruction execution.
- Multiplication operations can be executed in one clock cycle.
  - 16 bits x 16 bits → 32 bits
  - 32 bits x 32 bits → 32 bits or 64 bits
  - 32 bits x 32 bits → 64 bits
- Saturated operation instructions executed upon detection of an overflow or underflow
- 32-bit shift instructions executed in 1 clock cycle
- Bit manipulation instructions (SET1, CLR1, NOT1, TST1)
- Load and store instructions in long and short format
- Signed load instructions
- MAC operation instructions

  32 bits x 32 bits + 64 bits → 64 bits
- Floating-point operations

  FPU in the V850E2/Sx4-H complies with the ANSI/IEEE Standard for Binary Floating-Point Arithmetic (IEEE 754-1985).

3.1.1 On-chip peripherals

PPU base addresses

The addresses of the registers related to the peripheral protection features described in *V850E2M Architecture User's Manual* are defined here as offset addresses. The base address is:

<PPU_base> = FFFF 5100_H

PPU areas and registers

The following four registers are used to control each protection area (n = 0 to 4).

- PPVn – Used to enable general peripheral protection
- PPTn – Used to identify the type of general peripheral protection
- PPPn – Used to specify an OS peripheral
- PPSn – Used to specify a special peripheral

These registers contain 32 bits, which are named as follows (m = 0 to 31):

- PPVn.PPVnm
- PPTn.PPTnm
- PPPn.PPPnm
- PPSn.PPSnm

The range of protected addresses, the control registers and bits, and the name of each area are shown in *Table 3-1 "Areas and registers protected by the PPU"*.

Table 3-1 Areas and registers protected by the PPU (1/3)

Protection range	Protection control registers		Area name	Address ranges
	PPVn, PPTn, PPPn, PPSn registers n =	PPVnm, PPTnm, PPPnm, PPSnm bits m =		
1120 B	0	0	INTC	FFFF 6000_H to FFFF $645F_H$
256 B		22	MEMC	FFFF 7200_H to FFFF $72FF_H$
256 B		23	DMAC	FFFF 7300_H to FFFF $73FF_H$
256 B		24		FFFF 7400_H to FFFF $74FF_H$
512 B		25		FFFF 7500_H to FFFF $76FF_H$
512 B		28		FFFF $7B00_H$ to FFFF $7CFF_H$

Table 3-1 Areas and registers protected by the PPU (2/3)

Protection range	Protection control registers		Area name	Address ranges
	PPVn, PPTn, PPPn, PPSn registers n =	PPVnm, PPTnm, PPPnm, PPSnm bits m =		
64 KB	1	0	Port Pnm control	FF40 0000$_H$ to FF40 FFFF$_H$
		1	Port filter control	FF41 0000$_H$ to FF41 FFFF$_H$
		2	Clock generator Standby circuit Reset circuit	FF42 0000$_H$ to FF42 FFFF$_H$
		4	Port JPnm control	FF44 0000$_H$ to FF44 FFFF$_H$
		5	Shared with FE-level non-maskable interrupts	FF45 0000$_H$ to FF45 FFFF$_H$
		7	OCD	FF47 0000$_H$ to FF47 FFFF$_H$
		8	FCN0	FF48 0000$_H$ to FF49 FFFF$_H$
		9		FF49 0000$_H$ to FF48 FFFF$_H$
		10	FCN1	FF4A 0000$_H$ to FF4A FFFF$_H$
		11		FF4B 0000$_H$ to FF4B FFFF$_H$
		28	URTE0/LMA0	FF5C 0000$_H$ to FF5C FFFF$_H$
		29	URTE1/LMA1	FF5D 0000$_H$ to FF5D FFFF$_H$
		30	URTE2/LMA2	FF5E 0000$_H$ to FF5E FFFF$_H$
		31	URTE3/LMA3	FF5F 0000$_H$ to FF5F FFFF$_H$
	2	6	URTE10/LMA10	FF66 0000$_H$ to FF66 FFFF$_H$
		12	CSIH0	FF6C 0000$_H$ to FF6C FFFF$_H$
		13	CSIH1	FF6D 0000$_H$ to FF6D FFFF$_H$
		14	CSIH2	FF6E 0000$_H$ to FF6E FFFF$_H$
		16	CSIG0	FF70 0000$_H$ to FF70 FFFF$_H$
		20	CSIG4	FF74 0000$_H$ to FF74 FFFF$_H$
		22	Backup RAM	FF76 0000$_H$ to FF76 FFFF$_H$
		23	FE-level maskable interrupt selection, TAUA0 input selection, TAUJ0 input selection, FCN0 and FCN1 signal selection, IISA0 to IISA5, PCM0, and PCM1 signal selection, CSIH1 and CSIH2 signal selection	FF77 0000$_H$ to FF77 FFFF$_H$

Table 3-1 Areas and registers protected by the PPU (3/3)

Protection range	Protection control registers		Area name	Address ranges
	PPVn, PPTn, PPPn, PPSn registers n =	PPVnm, PPTnm, PPPnm, PPSnm bits m =		
4 KB	3	0	OSTM0	FF80 0000$_H$ to FF80 0FFF$_H$
		2	CLMA0	FF80 2000$_H$ to FF80 2FFF$_H$
		4	CLMA2	FF80 4000$_H$ to FF80 4FFF$_H$
		5	CLMA3	FF80 5000$_H$ to FF80 5FFF$_H$
		6	WDTA0	FF80 6000$_H$ to FF80 6FFF$_H$
		7	WDTA1	FF80 7000$_H$ to FF80 7FFF$_H$
		8	TAUA0	FF80 8000$_H$ to FF80 8FFF$_H$
		10	TAUB2	FF80 A000$_H$ to FF80 AFFF$_H$
		17	TAUJ0	FF81 1000$_H$ to FF81 1FFF$_H$
		20	RTCA0	FF81 4000$_H$ to FF81 4FFF$_H$
		25	ENCA0	FF81 9000$_H$ to FF81 9FFF$_H$
		26	ENCA1	FF81 A000$_H$ to FF81 AFFF$_H$
		29	ADCA0	FF81 D000$_H$ to FF81 DFFF$_H$
		31	DCRA0	FF81 F000$_H$ to FF81 FFFF$_H$
4 KB	4	0	IICB0	FF82 0000$_H$ to FF82 0FFF$_H$
		1	IICB1	FF82 1000$_H$ to FF82 1FFF$_H$
		2	IICB2	FF82 2000$_H$ to FF82 2FFF$_H$
		3	IICB3	FF82 3000$_H$ to FF82 3FFF$_H$
		8	IISA0 to IISA5	FF82 8000$_H$ to FF82 8FFF$_H$
		9	IISA BRG00 to BRG20, BRG01 to BRG21	FF82 9000$_H$ to FF82 9FFF$_H$
		11	KR0	FF82 B000$_H$ to FF82 BFFF$_H$
		12	IEBB0	FF82 C000$_H$ to FF82 CFFF$_H$
		13	MLB0	FF82 D000$_H$ to FF82 DFFF$_H$
		19	PCM0, PCM1	FF83 3000$_H$ to FF83 3FFF$_H$

3.1.2 Timing supervision features

TSU base address

The addresses of the registers related to the timing supervision features described in *V850E2M Architecture User's Manual* are defined here as offset addresses. The base address is:

<TSU_base> = FFFF 5000$_H$

3.2 Structure and Latency of CPU Access Bus

The CPU accesses the configuration, control and status registers of all modules of the microcontroller in different ways, depending on where they are located.

- CPU subsystem modules:

  For details, see *3.2.1 "Accessing CPU subsystem modules"*.

- Modules outside the CPU subsystem (PBUS and HBUS modules):

  For details, see *3.2.2 "Accessing the PBUS and HBUS modules"*.

Note For details about the CPU subsystem, see *3.3 "CPU Subsystem"*.

3.2.1 Accessing CPU subsystem modules

Two dedicated buses, LSPB and GSPB, are provided to access the registers of the modules in the CPU subsystem. These buses are controlled only by the CPU.

Table 3-2 Buses used to control V850E2M CPU subsystem

Module	CPU master bus	
	LSPB	GSPB
Interrupt controller (INTC)	R/W	–
Timing supervision unit (TSU)	R/W	–
Peripheral protection unit (PPU)	R/W	–
DMA controller (DMAC)	–	R/W
HBUS slave interface	–	R/W
HBUS master interface	–	R/W
PBUS interface	–	R/W
Data flash interface	–	R/W

Hardware locking The LSPB and GSPB buses can be hardware-locked. Bit manipulation instructions (CLR1, NOT1, SET1, and TST1) can therefore be used on all registers that can be accessed via the LSPB and GSPB buses and that can be accessed in 1-bit units.

Byte and halfword access The LSPB and GSPB buses support byte and halfword access. This means that byte and halfword data in registers that can be accessed by the LSPB and GSPB buses can be accessed separately.

3.2.2 Accessing the PBUS and HBUS modules

The CPU communicates with the PBUS and HBUS modules, which are not located on the CPU subsystem, by using the PBUS and HBUS master interfaces in the CPU subsystem.

Hardware locking

The PBUS and HBUS master interfaces cannot be hardware-locked. These modules therefore cannot be accessed in 1-bit units.

Byte and halfword access

The registers of the PBUS and HBUS modules are accessed in 32-bit units on word-aligned addresses, regardless of the size of the register. These registers cannot be accessed in byte or halfword units.

The clock supplied to the CPU subsystem is CKSCLK_000 (CPUCLK). When the CPU (or DMA controller) accesses a module outside the CPU subsystem, the bus clock might have to be synchronized with the CPU clock because modules outside the CPU subsystem operate on clocks supplied from different clock domains.

The bus structure, supplied clocks, and access latency when different bus clocks have to be synchronized are shown below.

Bus clock notation

The PCLK and HCLK clocks have an index that indicates which clock the bus is operating on. For example, $PCLK_{000}$ indicates PCLK synchronized with CKSCLK_000 supplied to PBUS. The following bus clock notation is generally used:

- $PCLK_{mn}$ is the PBUS clock synchronized with CKSCLK_mn.
- $HCLK_{mn}$ is the HBUS clock synchronized with CKSCLK_mn.

Caution

When a module in a clock domain that includes a bus synchronizer is accessed, a latency of several CPUCLK clock cycles occurs, which might cause inconsistencies in the flow of the CPU program.

For example, if an instruction that accesses a CPU subsystem module (specifically the DMAC or INTC) is issued after an instruction that accesses a module in which a latency has occurred due to the operation of the bus synchronizer, the access to the CPU subsystem module might be enabled first.

For details, see the description later in this chapter.

(1) Structure of CPU access bus

The structure of the bus when accessing other modules in the V850E2/Sx4-H device is shown in the figure below.

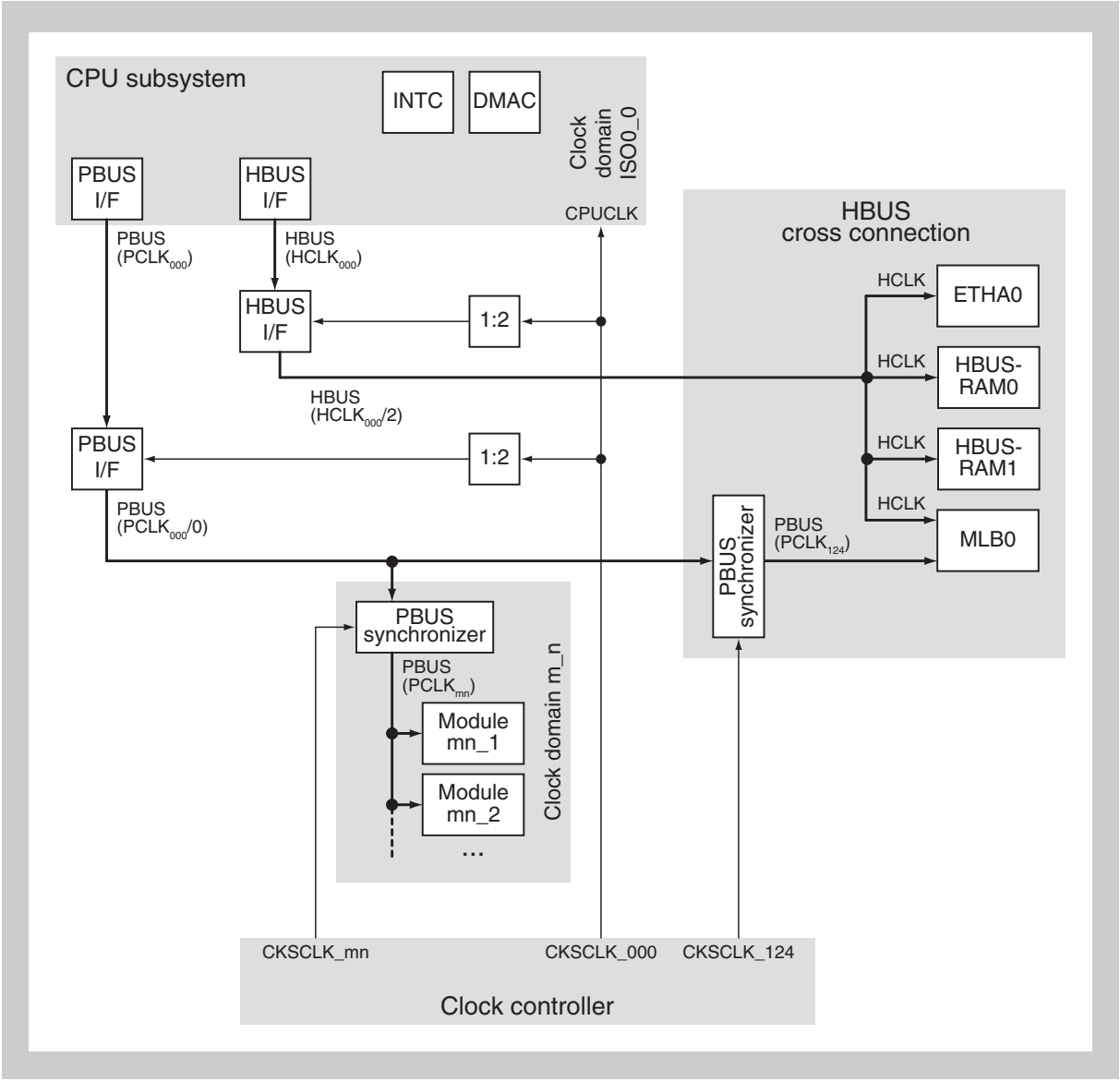


Figure 3-1 Structure of CPU access bus

The CPU accesses the following modules:

- The PBUS module operating on the clock $PCLK_{000}/2$
- The HBUS module operating on the clock $HCLK_{000}/2$

CPU subsystem modules

The CPU accesses CPU subsystem modules (such as the DMA controller and interrupt controller) by using the CPU subsystem bus. The clock supplied to the CPU subsystem bus is CPUCLK (CKSCLK_000). Bus clock synchronization is therefore not required when accessing modules in the CPU subsystem.

Clock domain m_n The clock supplied to the PBUS interface is supplied from a different clock domain (CKSCLK_mn).

Therefore, when the CPU accesses the PBUS interface, whose clock is supplied from clock domain m_n, by using the clock $PCLK_{000}/2$ = CKSCLK_000/2, this clock must be synchronized with the clock supplied from clock domain CKSCLK_mn.

For details about the latency caused by the PBUS synchronizer, see *3.2.3 "PBUS synchronizer"*.

Cross-connected HBUS modules The clock supplied to the HBUS interface for cross-connected HBUS modules is CKSCLK_000/2, which is already synchronized with the CPUCLK domain clock CKSCLK_000. A synchronizer is therefore not required.

3.2.3 PBUS synchronizer

The PBUS synchronizer is used to synchronize the bus clock for the CPU subsystem ($PCLK_{000}/2$) with the PBUS clock supplied from the target clock domain ($PCLK_{mn}$).

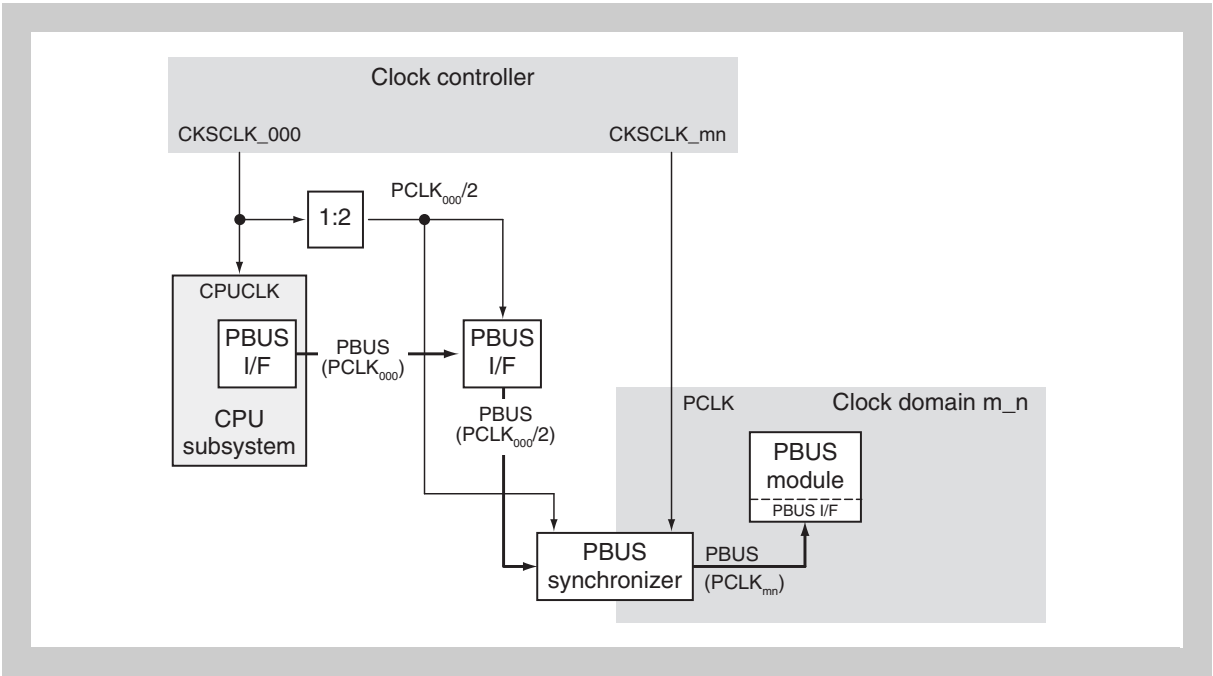


Figure 3-2 PBUS synchronizer

The synchronizer causes a latency in the access from the CPU to the PBUS module. The PBUS latency can be calculated by using the following equation:

Latency cycle = 4 x $PCLK_{000}/2$ + 4 x $PCLK_{mn}$

Note Synchronization always causes this latency, even if both clocks use the same clock source or have the same frequency.

3.3 CPU Subsystem

This section gives an overview of the CPU subsystem.

- CPU and CPU dedicated components (such as the processor protection circuits)
- Buses for instruction and data memories
- Interfaces for accessing other microcontroller peripherals (such as PBUS and the data flash interface)
- Interrupt controller (INTC)
- DMA controller (DMAC)
- On-chip debugger (OCD)
- Multiple bus systems that can be used to allow the CPU to access all other peripherals as the bus master

3.3.1 Power supply and clock domain

The CPU subsystem is assigned to Isolated area 0 and the CPU clock (CPUCLK) is supplied based on the CKSCLK_000 clock.

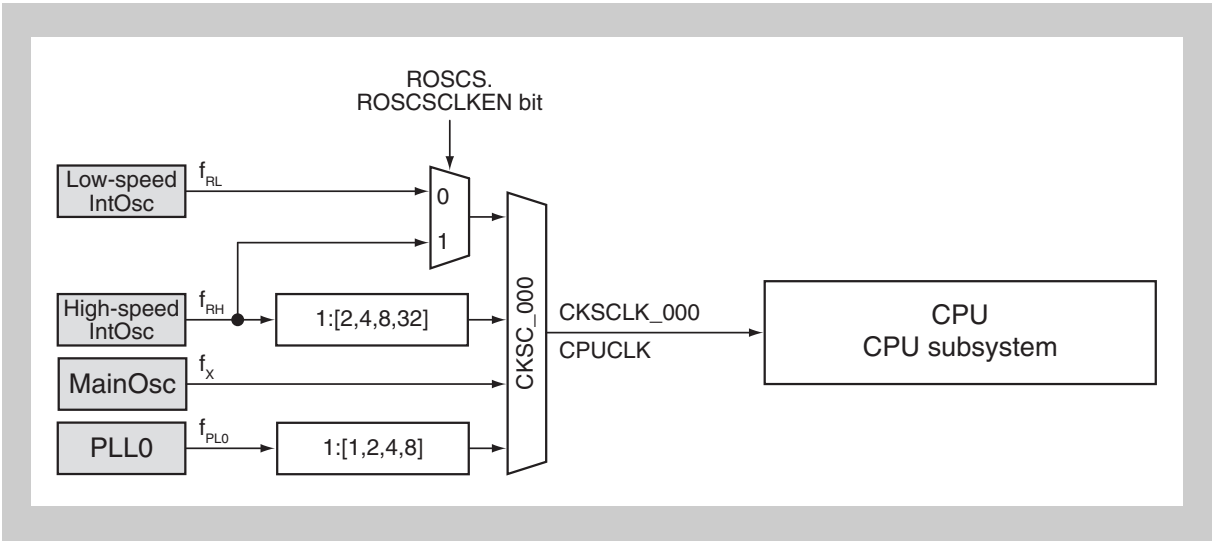


Figure 3-3 CPU subsystem clock supply

CPU subsystem hardware reset

The CPU subsystem and its registers are initialized by the following reset signal:

Table 3-3 CPU subsystem reset signal

CPU subsystem	Reset signal
CPU subsystem	System reset SYSRES

3.3.2 Overview of CPU subsystem buses

An overview of the CPU subsystem buses and their general applications is given below.

For details about the CPU subsystem, see the next section.

Table 3-4 Bus systems

CPU subsystem	Bus	Application	Bus master
Internal	Local system peripheral bus LSPB	For accessing the following peripheral registers: • Timing supervision unit (TSU) • Peripheral protection unit (PPU) • Interrupt controller (INTC)	CPU
	CPU system bus	For the CPU to access the following components: • External HBUS slave (via the HBUS master interface) • PBUS/GSPB bus • Data flash	CPU
	DMA data bus	For the DMAC to access HBUS • Data RAM • External HBUS slave (via the HBUS master interface) • PBUS/GSPB bus • Data flash	DMAC
	HBUS data bus	For the external HBUS master to access the following components (via the HBUS slave interface): • Data RAM • Code flash • External HBUS slave (via the HBUS master interface) • PBUS/GSPB bus • Data flash	External HBUS master (via the HBUS slave interface)
	Global system peripheral bus (GSPB)	For accessing the following peripheral registers: • DMA controller (DMAC) • HBUS bridge • PBUS interface • Data flash interface	CPU, DMAC, external HBUS master (via the HBUS slave interface)
External	PBUS	For accessing the PBUS module (via the PBUS master interface)	CPU, DMAC, external HBUS master (via the HBUS slave interface)[a]
	HBUS	For accessing the HBUS slave module (via the HBUS master interface)	CPU, DMAC, external HBUS master (via the HBUS slave interface)[a]

a) For details about bus arbitration, see the next section.

3.3.3 V850E2/Sx4-H CPU subsystem

A block diagram of the CPU subsystem in the V850E2/Sx4-H is shown below.

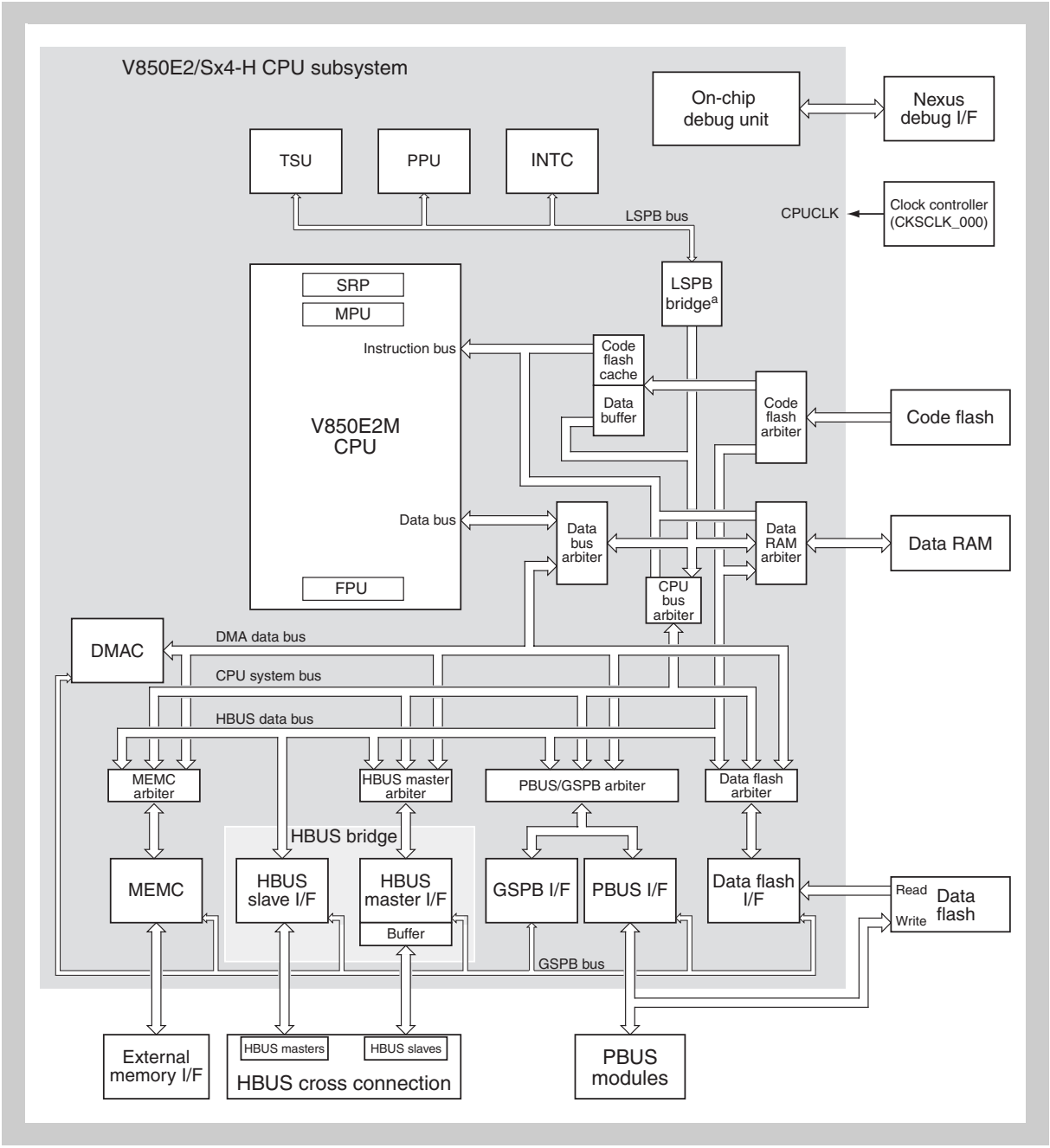


Figure 3-4 V850E2/Sx4-H CPU subsystem

a) Accessing the LSPB bus via the LSPB bridge is only possible when the CPU is accessing the CPU data bus via the data bus arbiter.

The buses that transfer data between modules are controlled by the following three masters:

- CPU
- DMA controller (DMAC)
- HBUS master (via the HBUS slave interface)

Table 3-5 Data and instruction buses in the CPU subsystem of the V850E2/Sx4-H

Master	Bus	Bus width	Code flash	Data RAM	Data flash	PBUS/ GSPB interface	HBUS master interface	MEMC
CPU	Instruction	128 bits	R	R[a]	R[a]	R[a]	R[a]	R[a]
	Data	32 bits	R	R/W	R/W	R/W	R/W	R/W
DMA controller	DMA data bus	32 bits	R	R/W	R	R/W	R/W	R/W
HBUS slave interface	HBUS data bus	32 bits	R	R/W	R	R/W	R/W	R/W

a) CPU instruction fetches from these sources tend to have a heavy CPU clock cycle occupancy, lowering the CPU performance. We therefore do not recommend executing CPU instruction fetches from these sources.

(1) Accessing the code flash area

Table 3-6 Accessing the code flash area

<table>
<tr><td rowspan="3">Code flash</td><td rowspan="3">Code flash/ data flash</td><td colspan="2">← CPU instruction bus</td></tr>
<tr><td rowspan="2">Data bus arbiter</td><td>← CPU data bus</td></tr>
<tr><td>← DMA data bus</td></tr>
<tr><td colspan="2">← HBUS data bus</td></tr>
</table>

(2) Accessing data RAM

Caution Before fetching any instruction code from the data RAM, make sure to initialize the 16-byte boundary area of the data RAM that contains the instruction code to fetch.

A 16-byte boundary area is an area from the address XXXX XXX0H to the address XXXX XXXFH.

Any data values can be written when initializing the data RAM, but the data RAM area must be initialized before an instruction is fetched from it.

If an instruction is fetched from an uninitialized data RAM area, a memory protection exception (MEP) might occur.

Note In general, it is recommended to initialize all the data RAM before reading it.

Table 3-7 Accessing the data RAM and backup RAM

<table>
<tr><td rowspan="4">Data RAM</td><td rowspan="4">Data RAM arbiter</td><td colspan="2">← CPU instruction bus</td></tr>
<tr><td rowspan="2">Data bus arbiter</td><td>← CPU data bus</td></tr>
<tr><td>← DMA data bus</td></tr>
<tr><td colspan="2">← HBUS data bus</td></tr>
</table>

(3) Accessing the data flash interface

Table 3-8 Accessing the data flash interface

<table>
<tr><td rowspan="4">Data flash interface</td><td rowspan="4">Data flash arbiter</td><td rowspan="3">CPU Bus arbiter</td><td colspan="2">← CPU instruction bus</td></tr>
<tr><td rowspan="2">Data bus arbiter</td><td>← CPU data bus</td></tr>
<tr><td>X (direct DMA access via the data flash arbiter)</td></tr>
<tr><td colspan="3">← HBUS data bus</td></tr>
<tr><td colspan="3">← DMA data bus</td></tr>
</table>

(4) Accessing the PBUS/GSPB interface

Table 3-9 Accessing the PBUS/GSPB interface

<table>
<tr><td rowspan="4">PBUS/GSPB interface</td><td rowspan="4">PBUS/ GSPB arbiter</td><td rowspan="3">CPU Bus arbiter</td><td colspan="2">← CPU instruction bus</td></tr>
<tr><td rowspan="2">Data bus arbiter</td><td>← CPU data bus</td></tr>
<tr><td>X (direct DMA access via the PBUS/GBUS arbiter)</td></tr>
<tr><td colspan="3">← HBUS data bus</td></tr>
<tr><td colspan="3">← DMA data bus</td></tr>
</table>

(5) Accessing the HBUS master interface

Table 3-10 Accessing the HBUS master interface

<table>
<tr><td rowspan="4">HBUS master interface</td><td rowspan="4">HBUS master arbiter</td><td rowspan="3">CPU Bus arbiter</td><td colspan="2">← CPU instruction bus</td></tr>
<tr><td rowspan="2">Data bus arbiter</td><td>← CPU data bus</td></tr>
<tr><td>X (direct DMA access via the HBUS master arbiter)</td></tr>
<tr><td colspan="3">← HBUS data bus</td></tr>
<tr><td colspan="3">← DMA data bus</td></tr>
</table>

(6) Accessing the MEMC

Table 3-11 Accessing the MEMC

<table>
<tr><td rowspan="4">MEMC</td><td rowspan="4">MEMC arbiter</td><td rowspan="3">CPU bus arbiter</td><td colspan="2">← CPU instruction bus</td></tr>
<tr><td rowspan="2">Data bus arbiter</td><td>← CPU data bus</td></tr>
<tr><td>X (direct DMA access via the MEMC arbiter)</td></tr>
<tr><td colspan="3">← HBUS data bus</td></tr>
<tr><td colspan="3">← DMA data bus</td></tr>
</table>

(7) Arbitration between the CPU subsystem buses

How arbitration between the CPU subsystem buses works is shown in the table below.

Table 3-12 Arbitration via single-stage arbiter

Arbiter	Policy	Master
Code flash cache/data buffer	Fixed priority	High: DMAC/CPU data bus (via data bus arbiter)
		Low: CPU instruction bus
Code flash	Round robin	HBUS
		Code flash cache/data buffer
Data RAM	Round robin	CPU instruction bus
		CPU data bus/DMA (via data bus arbiter)
		HBUS
Data bus	Fixed priority	High: DMA
		Low: CPU data bus
CPU bus	Fixed priority	High: DMAC/CPU data bus (via data bus arbiter)
		Low: CPU instruction bus

Table 3-13 Arbitration via two-stage arbiter

Arbiter	Policy	Master
HBUS master PBUS/GSPB data flash	First stage:	
	Round robin	HBUS
		CPU system bus (from CPU bus arbiter)
	Second stage:	
	Fixed priority	High: DMA
		Low: HBUS/CPU system bus (from first stage)

(8) HBUS bridge in CPU subsystem

For details about the HBUS bridge in the CPU subsystem, see *3.9 "HBUS Bridge in CPU Subsystem" on page 187*.

3.3.4 User's manuals describing the V850E2M system

For details about the features of the CPU subsystem, see the following user's manuals:

Table 3-14 Sources of information about CPU subsystem features

Feature	V850E2M Architecture User's Manual (R01US001E)	This manual
V850E2M CPU (including instruction set)	√	–
Floating-point unit (FPU)	√	–
Processor protection features (MPU, SPR, PPU, TPU)	√	–
DMA controller (DMAC)	–	√
Interrupt controller (INTC)	–	√
HBUS bridge	–	√
Code flash/Data flash	–	√
Data RAM/backup RAM	–	√

3.4 HBUS Cross-Connection System

The following modules are connected to each other in the HBUS cross-connection system.

- CPU subsystem
- Ethernet controller (ETHA0)
- Medial local bus (MLB)
- HBUS-RAM0
- HBUS-RAM1

(1) HBUS clock supply

The clock supplied to the HBUS cross-connection system and related modules is the CKSC_000 CPU subsystem domain clock divided by two. It has a maximum frequency of 80 MHz.

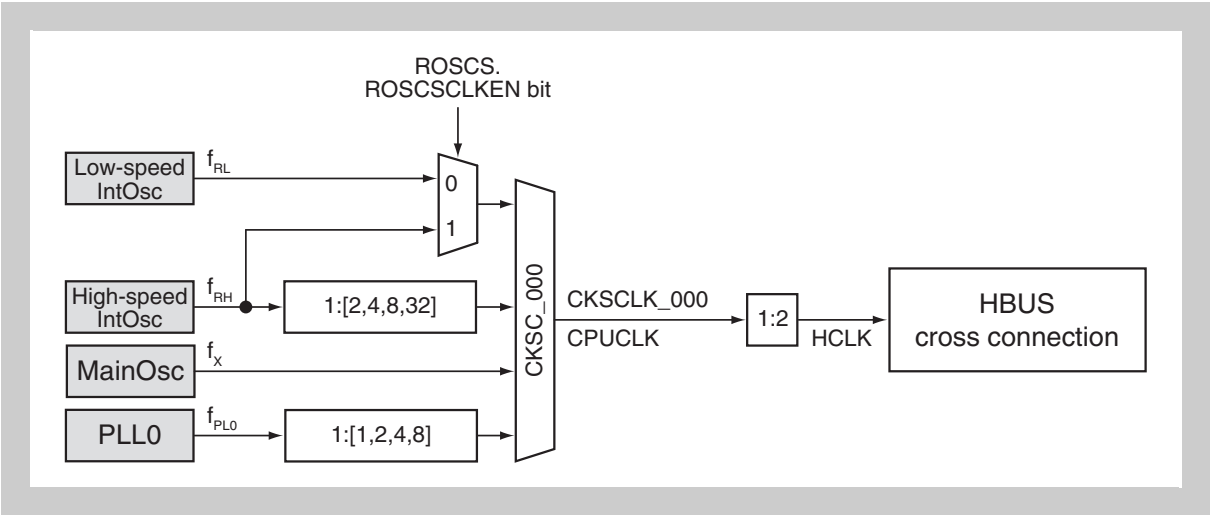


Figure 3-5 HBUS clock supply

(2) HBUS overview

The HBUS cross-connection system allows slave modules to be accessed by multiple bus masters.

The modules connected to HBUS and their data port attributes are shown below.

Notes
1. The figure below does not include the ports used to control access to the registers of each module.
2. The arrows in the figure indicate the direction in which the data flows.

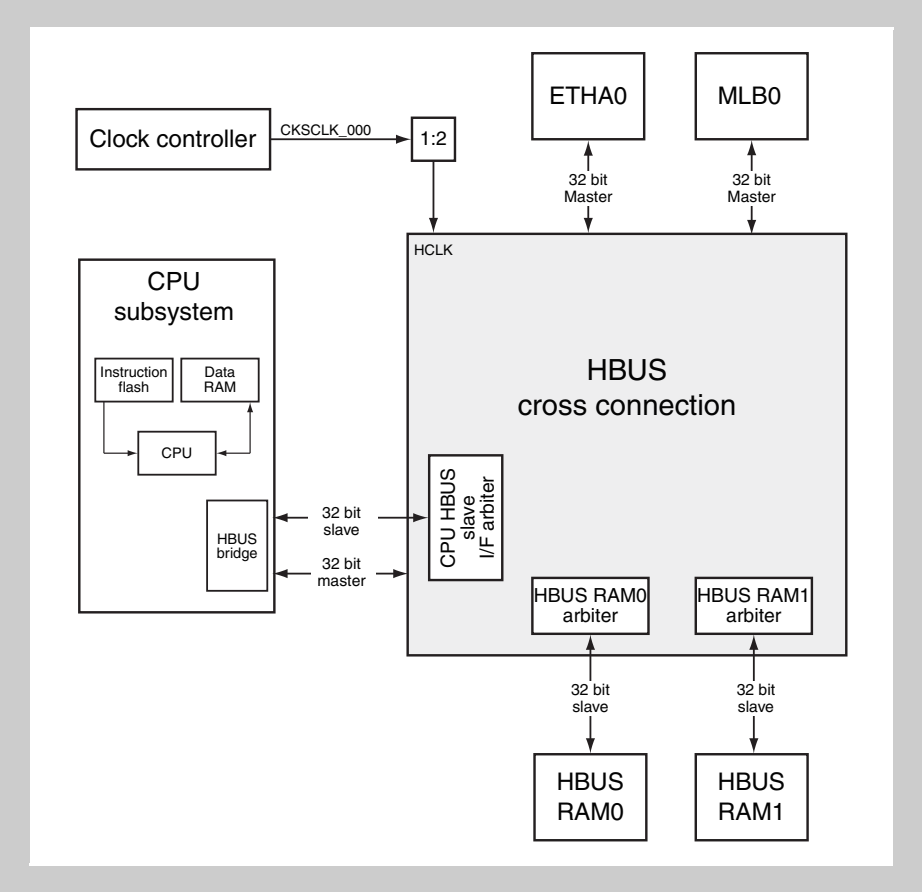


Figure 3-6 HBUS cross-connection system

(3) HBUS data ports

The HBUS data ports are used to transfer data between the modules on HBUS. The HBUS ports are described in the following tables.

Table 3-15 HBUS master data ports

Master ports	Data width
Ethernet controller data ports	32 bits
Media local bus data ports	32 bits
Data ports of HBUS master interface in CPU subsystem	32 bits

Table 3-16 HBUS slave data ports

Slave ports	Data width
HBUS-RAM0 data ports	32 bits
HBUS-RAM1 data ports	32 bits
Data ports of HBUS slave interface in CPU subsystem	32 bits

3.4.1 HBUS cross-connection matrix

A block diagram of the bus used to control the transfer of data from master to slave is shown below.

Note The arrows in the figure indicate the access direction (not the direction in which the data flows).

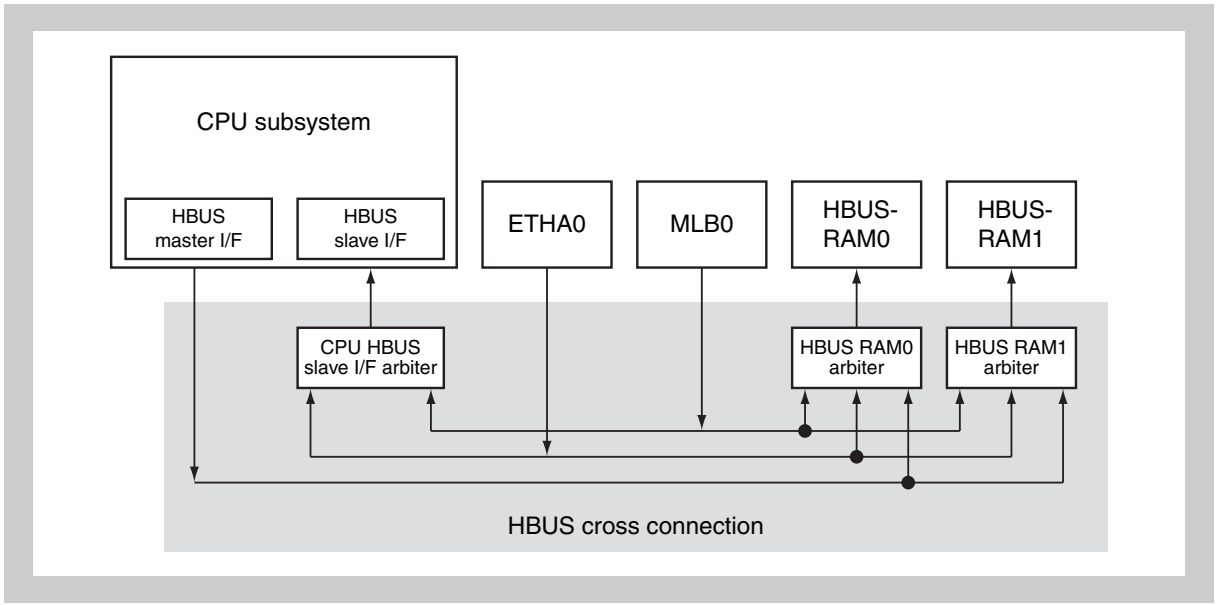


Figure 3-7 HBUS control bus

In the HBUS cross-connection system, bus masters can be connected to slaves as shown in the table below.

Table 3-17 HBUS data path cross-connection matrix

Master data ports	Slave data ports		
	HBUS-RAM0	HBUS-RAM1	HBUS slave interface in CPU subsystem
Ethernet controller data ports	R/W	R/W	R/W
Media local bus data ports	R/W	R/W	R/W
Data ports of HBUS master interface in CPU subsystem	R/W	R/W	–

3.4.2 HBUS arbitration policy

The following HBUS arbiters perform arbitration using the round robin system:

- HBUS-RAM0 arbiter
- HBUS-RAM1 arbiter
- Arbiter for HBUS slave interface in CPU subsystem

3.5 Operating Modes

This section describes the operating modes of the V850E2/Sx4-H and how they are selected.

The V850E2/Sx4-H provides the following operating modes:

- Normal mode
- Flash programming mode

After reset is cancelled, the microcontroller starts to fetch instructions from an internal boot ROM which contains the internal firmware. The firmware checks the FLMD0 pin, and optionally also the FLMD1 pin, to set the operation mode after reset is cancelled according to the table below.

When using the FLMD1 pin in the normal mode, be sure to set its input level according to the table below after reset is canceled.

Note When using the FLMD0 pin, be sure to connect it to E0VSS level via a pull-down resistor with a resistance of at least 82 kΩ.

Table 3-18 Operating mode selection

Pin		Operating mode
FLMD0	FLMD1 (P0_1)	
L	L	Normal mode
	H	Setting prohibited
H	L	Flash programming mode

Note L: Low-level input, H: High-level input

3.5.1 Normal mode

In this mode, the user-created program is executed.

The on-chip flash memory cannot be reprogrammed in this mode.

After reset is cancelled, the firmware branches to the start of the active boot swap cluster.

3.5.2 Flash programming mode

In flash programming mode, the on-chip flash memory is erased and reprogrammed.

After reset is cancelled, the firmware initiates loading of the user's program code from the external flash programmer and programs the flash memory.

After disconnecting the external flash programmer, the microcontroller can be started up by the new user's program in normal mode.

For details, see *Chapter 7 “Flash Memory”*.

3.6 Address Space

The address space of the CPU (including the size and addresses of the physical address space) is described in the following sections.

The address range of the data space and program space are described in terms of their wraparound properties.

3.6.1 CPU data address space and physical program address space

The CPU supports the following address spaces:

- 4 GB CPU data address space

  Addresses for a 4 GB memory can be generated by using the 32-bit general purpose registers. This is the maximum address space supported by the CPU.

- 512 MB physical program address space

  The CPU provides a 512 MB physical address space to access instruction code in the program memory. This means that an internal or external program memory of up to 512 MB can be accessed.

3.6.2 Program space and data space

The figure below shows the assignment of the data and program spaces in the CPU address space.

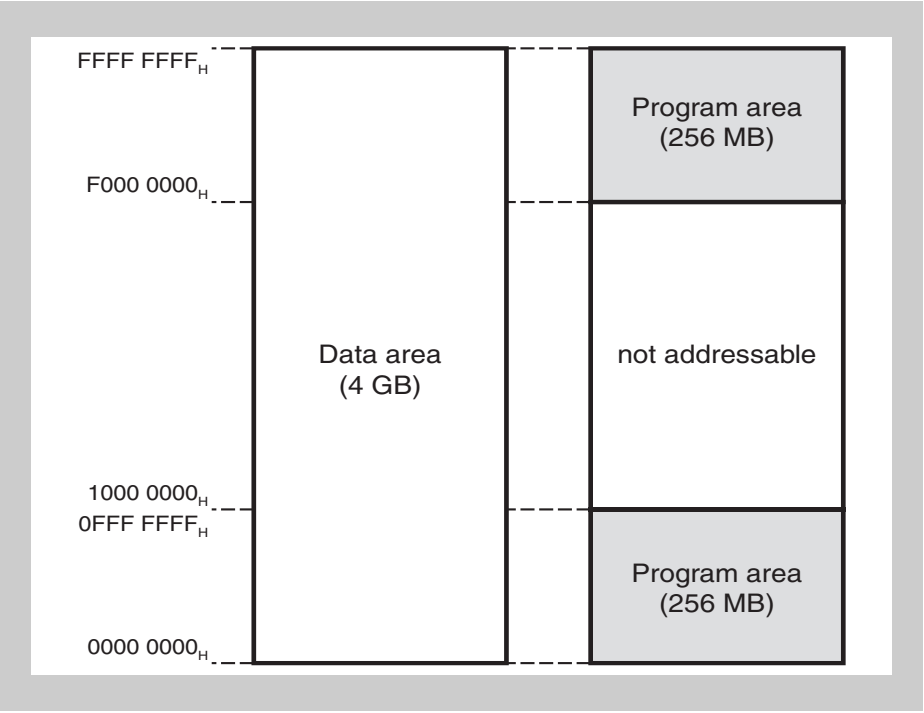


Figure 3-8 CPU address space

(1) Data space wraparound

If an operand address calculation exceeds 32 bits, the exceeding bits are ignored.

Therefore, the addresses FFFF FFFF$_H$, which indicates the upper limit of the data space, and 0000 0000$_H$, which indicates the lower limit, are contiguous addresses, which results in a wraparound of the data space at this boundary.

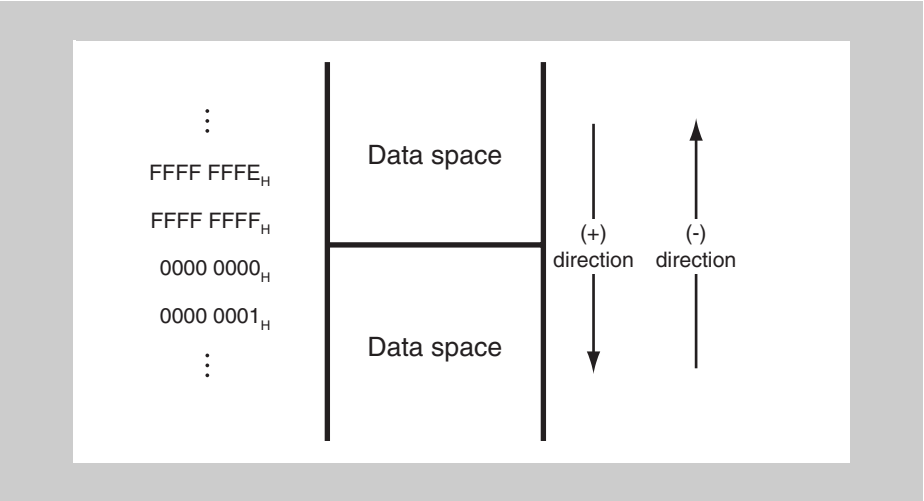


Figure 3-9 Data space wraparound

(2) Program space wraparound

If an instruction address calculation exceeds 28 bits, the exceeding bits are ignored.

Therefore, the addresses 0FFF FFFF$_H$, which indicates the upper limit of the data space, and 0000 0000$_H$, which indicates the lower limit, are contiguous addresses, which results in a wraparound of the data space at this boundary.

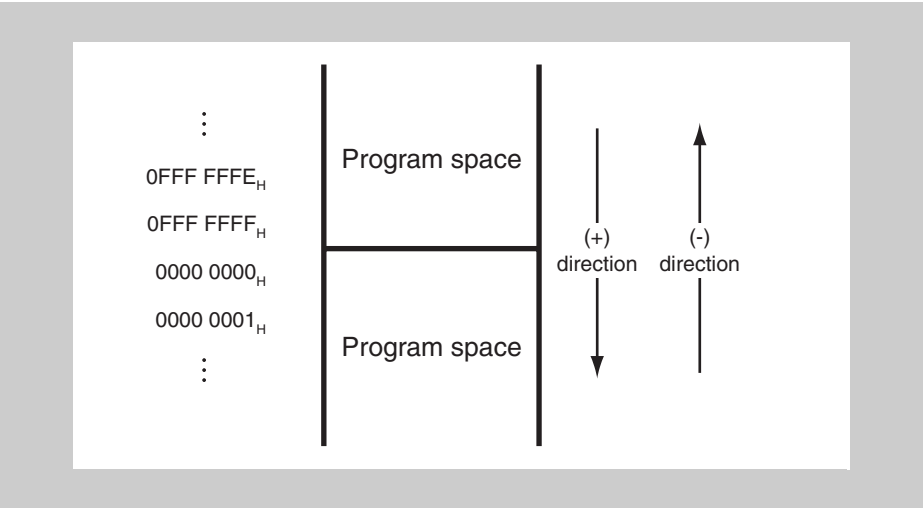


Figure 3-10 Program space wraparound

3.7 CPU Address Map

This section describes the memory map of the CPU.

3.7.1 DMAC address map

The DMA controller can access all CPU address areas except the following:

FFFF 5000_H to FFFF $7FFF_H$

This area can only be accessed by the CPU.

3.7.2 Memory map

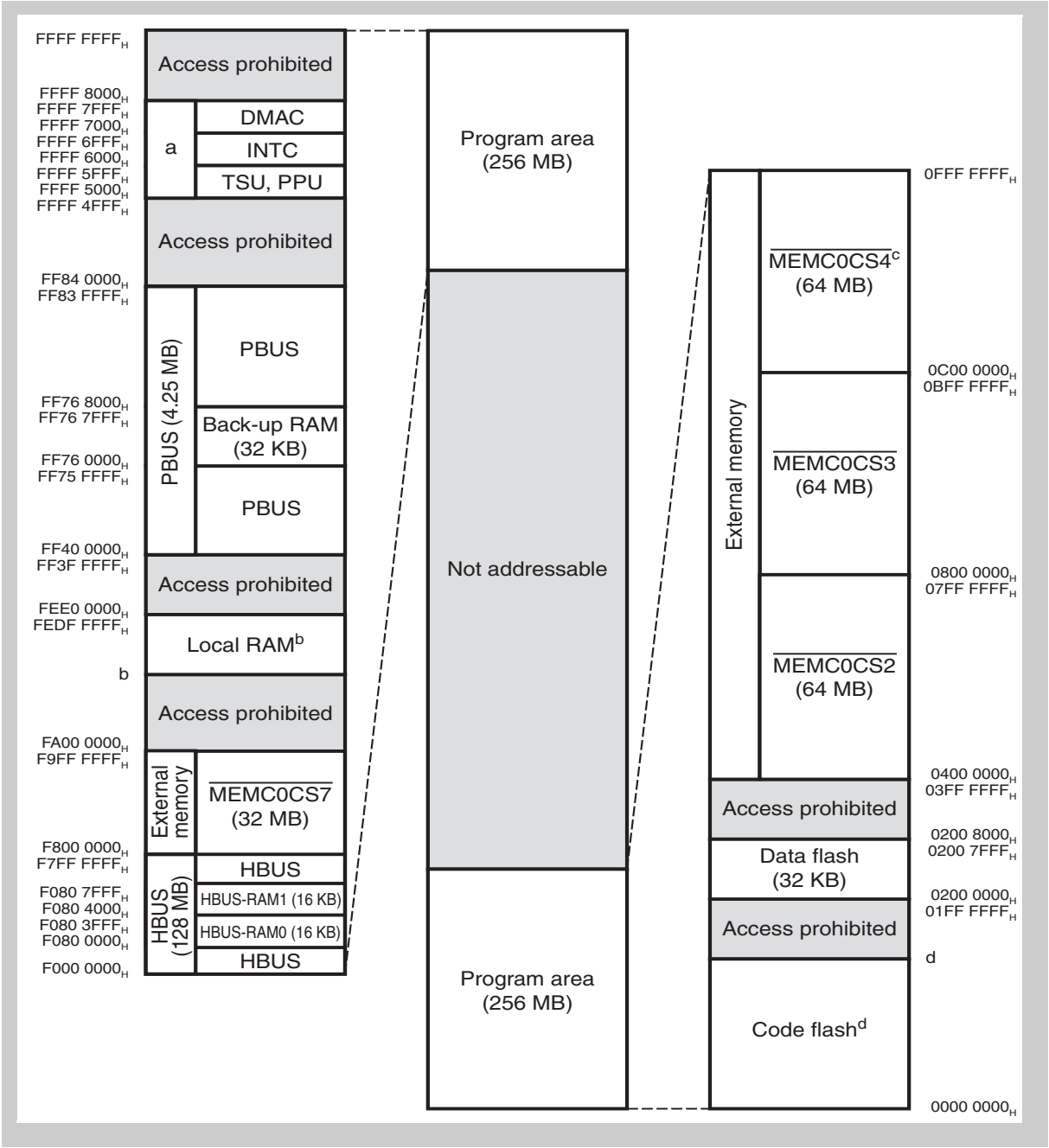


Figure 3-11 Memory map

a) This area cannot be accessed by the DMA controller.

b) The local RAM size and its address range differ depending on the device. For details, see *Table 3-21 "Internal local RAM area"*.

c) Only $\overline{\text{MEMC0CS4}}$ can be connected to SDRAM.

d) The code flash size and its address range differ depending on the device. For details, see *Table 3-19 "Code flash memory area"*.

3.7.3 Memory areas

The following areas are assigned as internal memory areas in the V850E2/Sx4-H:

- Internal code flash area
- Internal data flash area
- Internal local RAM area
- Internal backup RAM area
- Internal HBUS-RAM area
- External memory area

Each of these areas is described below.

(1) Internal code flash area

The sizes and addresses of the code flash memories are described in the table below.

Table 3-19 Code flash memory area

Name	Part number	Size	Address range
SG4-H-1M	μPD70F4013	1 MB	0000 0000_H to 000F $FFFF_H$
SG4-H-1.5M	μPD70F4014	1.5 MB	0000 0000_H to 0017 $FFFF_H$
SJ4-H-1M	μPD70F4015	1 MB	0000 0000_H to 000F $FFFF_H$
SJ4-H-1.5M	μPD70F4016	1.5 MB	0000 0000_H to 0017 $FFFF_H$
SK4-H-1.5M	μPD70F4017	1.5 MB	0000 0000_H to 0017 $FFFF_H$
SK4-H-2M	μPD70F4018	2 MB	0000 0000_H to 001F $FFFF_H$

(2) Internal data flash area

The sizes and addresses of the data flash memories are described in the table below.

Table 3-20 Internal data flash area

Name	Part number	Size	Address range
SG4-H-1M	μPD70F4013	32 KB	0200 0000_H to 0200 $7FFF_H$
SG4-H-1.5M	μPD70F4014	32 KB	0200 0000_H to 0200 $7FFF_H$
SJ4-H-1M	μPD70F4015	32 KB	0200 0000_H to 0200 $7FFF_H$
SJ4-H-1.5M	μPD70F4016	32 KB	0200 0000_H to 0200 $7FFF_H$
SK4-H-1.5M	μPD70F4017	32 KB	0200 0000_H to 0200 $7FFF_H$
SK4-H-2M	μPD70F4018	32 KB	0200 0000_H to 0200 $7FFF_H$

(3) Internal local RAM area

The sizes and addresses of the local RAMs are described in the table below.

Table 3-21 Internal local RAM area

Name	Part number	Size	Address range
SG4-H-1M	μPD70F4013	96 KB	FEDE 8000_H to FEDF $FFFF_H$
SG4-H-1.5M	μPD70F4014	128 KB	FEDE 0000_H to FEDF $FFFF_H$
SJ4-H-1M	μPD70F4015	96 KB	FEDE 8000_H to FEDF $FFFF_H$
SJ4-H-1.5M	μPD70F4016	128 KB	FEDE 0000_H to FEDF $FFFF_H$
SK4-H-1.5M	μPD70F4017	128 KB	FEDE 0000_H to FEDF $FFFF_H$
SK4-H-2M	μPD70F4018	192 KB	FEDD 0000_H to FEDF $FFFF_H$

<R> **Caution** Before fetching any instruction code from the data RAM, make sure to initialize the 16-byte boundary area of the data RAM that contains the instruction code to fetch.
A 16-byte boundary area is an area from address XXXX XXX0H to address XXXX XXXFH.
For initializing the data RAM, any data values can be written, but the data RAM area must be initialized before an instruction is fetched from it. If an instruction is fetched from an uninitialized data RAM area, a memory protection exception (MEP) might occur.

<R> **Note** It is recommended to initialize the entire data RAM before reading data from the data RAM.

(4) Internal backup RAM area

The sizes and addresses of the backup RAMs are described in the table below.

Table 3-22 Backup RAM area

Name	Part number	Size	Address range
SG4-H-1M	μPD70F4013	32 KB	FF76 0000_H to FF76 $7FFF_H$
SG4-H-1.5M	μPD70F4014	32 KB	FF76 0000_H to FF76 $7FFF_H$
SJ4-H-1M	μPD70F4015	32 KB	FF76 0000_H to FF76 $7FFF_H$
SJ4-H-1.5M	μPD70F4016	32 KB	FF76 0000_H to FF76 $7FFF_H$
SK4-H-1.5M	μPD70F4017	32 KB	FF76 0000_H to FF76 $7FFF_H$
SK4-H-2M	μPD70F4018	32 KB	FF76 0000_H to FF76 $7FFF_H$

<R> **Note** Instructions cannot be fetched from the backup RAM area. The back-up RAM area can be read or written only in 32-bit units.

Writing to the backup RAM The backup RAM must be written by using a specific procedure. For details, see *3.8.1 "Protecting the backup RAM" on page 185*.

(5) Internal HBUS-RAM area

The sizes and addresses of the HBUS-RAM are described in the table below.

Table 3-23 HBUS-RAM area

Name	Part number	Size	Address range
SG4-H-1M	μPD70F4013	HBUS-RAM0: 16 KB	F080 0000$_H$ to F080 3FFF$_H$
		HBUS-RAM1: 16 KB	F080 4000$_H$ to F080 7FFF$_H$
SG4-H-1.5M	μPD70F4014	HBUS-RAM0: 16 KB	F080 0000$_H$ to F080 3FFF$_H$
		HBUS-RAM1: 16 KB	F080 4000$_H$ to F080 7FFF$_H$
SJ4-H-1M	μPD70F4015	HBUS-RAM0: 16 KB	F080 0000$_H$ to F080 3FFF$_H$
		HBUS-RAM1: 16 KB	F080 4000$_H$ to F080 7FFF$_H$
SJ4-H-1.5M	μPD70F4016	HBUS-RAM0: 16 KB	F080 0000$_H$ to F080 3FFF$_H$
		HBUS-RAM1: 16 KB	F080 4000$_H$ to F080 7FFF$_H$
SK4-H-1.5M	μPD70F4017	HBUS-RAM0: 16 KB	F080 0000$_H$ to F080 3FFF$_H$
		HBUS-RAM1: 16 KB	F080 4000$_H$ to F080 7FFF$_H$
SK4-H-2M	μPD70F4018	HBUS-RAM0: 16 KB	F080 0000$_H$ to F080 3FFF$_H$
		HBUS-RAM1: 16 KB	F080 4000$_H$ to F080 7FFF$_H$

(6) External memory area

The sizes and addresses of the external memories are described in the table below.

Caution In the V850E2/SG4-H, signals are not output from the $\overline{\text{MEMC0CS2}}$, $\overline{\text{MEMC0CS3}}$, $\overline{\text{MEMC0CS4}}$, and $\overline{\text{MEMC0CS7}}$ pins. In the V850E2/SJ4-H, signals are not output from the $\overline{\text{MEMC0CS7}}$ pin.

Table 3-24 External memory area

Name	Size	Address range
V850E2/Sx4-H	64 MB	$\overline{\text{MEMC0CS2}}$: 0400 0000$_H$ to 07FF FFFF$_H$
	64 MB	$\overline{\text{MEMC0CS3}}$: 0800 0000$_H$ to 0BFF FFFF$_H$
	64 MB	$\overline{\text{MEMC0CS4}}$: 0C00 0000$_H$ to 0FFF FFFF$_H$
	32 MB	$\overline{\text{MEMC0CS7}}$: F800 0000$_H$ to F9FF FFFF$_H$

3.8 Backup RAM (BURAM)

The 16 KB backup RAM is a PBUS module configured as 32 bits × 4 K.

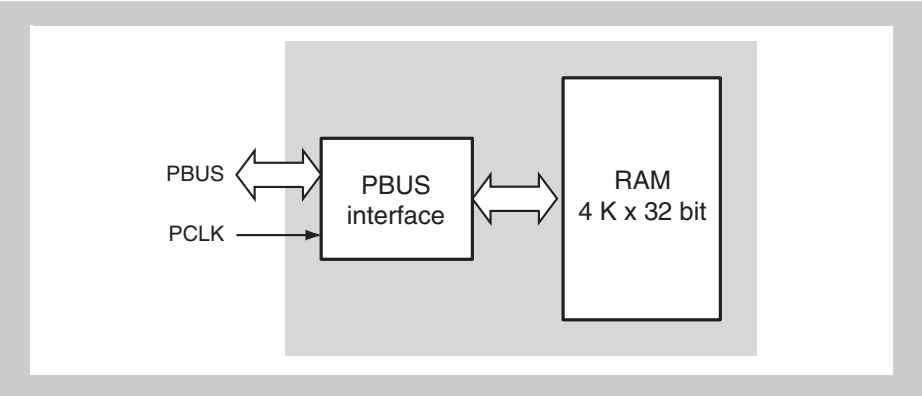


Figure 3-12 Block diagram of backup RAM

Access The backup RAM can be read or written in 32-bit units.

BURAM addresses The BURAM addresses are shown in the following table.

Table 3-25 BURAM addresses

BURAM	Address
BURAM	FF76 0000_H to FF76 $7FFF_H$

Clock supply The following clock is supplied to BURAM:

Table 3-26 BURAM clock supply

BURAM	BURAM clock	Connected to:
BURAM	PCLK	Clock controller CKSCLK_A05

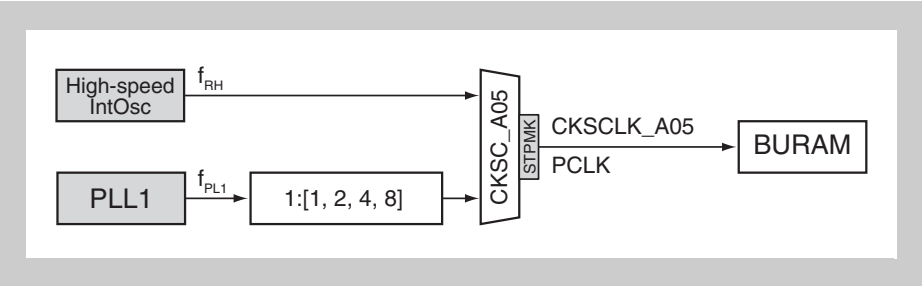


Figure 3-13 BURAM clock supply

Writing to BURAM Writing to the backup RAM must be explicitly enabled by using the backup RAM control register (BURC).

3.8.1 Protecting the backup RAM

Writing to the backup RAM is prohibited immediately after reset is cancelled.

Writing to the backup RAM must therefore be explicitly enabled by setting the write-enable bit (BURC.BURWE) to 1.

If the backup RAM is written while writing is prohibited (while the BURC.BURWE bit is 0), an error bit (BURAE.BURAERR) is set.

The following registers control and monitor write accesses to the backup RAM:

Table 3-27 List of backup RAM registers

Register name	Symbol	Address
Backup RAM control register	BURC	FF76 FE00$_H$
Backup RAM write access error register	BURAE	FF76 FE04$_H$
Backup RAM write access error clear register	BURAEC	FF76 FE08$_H$

(1) BURC – Backup RAM control register

The BURC register is used to enable and disable writing to the backup RAM.

Access This register can be read or written in 8-bit or 1-bit units.

Address FF76 FE00$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	BURWE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-28 BURC register contents

Bit position	Bit name	Function
0	BURWE	This bit enables and disables writing to the backup RAM. 0: Writing to the backup RAM is disabled. 1: Writing to the backup RAM is enabled.

(2) BURAE – Backup RAM access error register

The BURAE register monitors whether an error occurred when writing to the backup RAM.

Access This register is read-only, in 1-bit or 8-bit units.

Address FF76 FE04$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	BURAERR
R	R	R	R	R	R	R	R

Table 3-29 BURAE register contents

Bit position	Bit name	Function
0	BURAERR	Backup RAM write access error flag 0: No backup RAM write access error occurred. 1: A backup RAM write access error occurred.

(3) BURAEC – Backup RAM access error clear register

The BURAEC register clears the backup RAM write access error flag (BURAERR).

Access This register can be read or written in 1-bit or 8-bit units.

Address FF76 FE08$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	BURAERRC
R	R	R	R	R	R	R	R/W

Table 3-30 BURAEC register contents

Bit position	Bit name	Function
0	BURAERRC	This bit clears the backup RAM write access error flag (BURAERR). 0: No operation 1: BURAERR cleared

3.9 HBUS Bridge in CPU Subsystem

The CPU subsystem HBUS bridge provides an interface between the internal resources of the CPU subsystem and the HBUS module outside the CPU subsystem.

HBUS slave interface

The HBUS slave interface connects the CPU to HBUS masters outside the CPU subsystem, giving them access to the internal resources of the CPU subsystem.

HBUS master interface

The HBUS master interface connects the CPU to HBUS slaves outside the CPU subsystem, giving modules in the CPU subsystem (such as the CPU and DMAC) access to the HBUS resources outside the CPU subsystem. The CPU

<R> subsystem accesses external HBUS slaves via buffers.

All HBUS masters and slaves outside the CPU subsystem are connected by the HBUS cross-connection system.

<R>

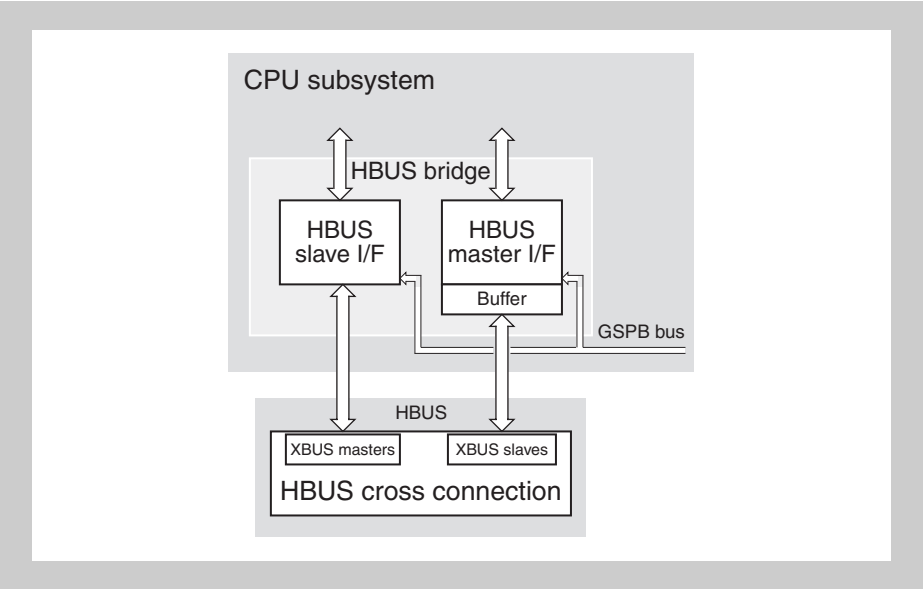


Figure 3-14 HBUS bridge in CPU subsystem

3.9.1 Description

(1) HBUS master interface (for transferring data from the CPU subsystem to HBUS)

Transfer type

Data can be sent from the CPU subsystem to areas covered by the HBUS bridge between the CPU subsystem and the HBUS module in the following ways:

- Direct access
- Buffer access

<R>

Memory areas

Four memory areas can be defined by specifying the base address and size of the memory. The type of transfer to be used for each area can also be defined separately. These settings are specified by using the ETACFG, ETACFGn, and ETARADRSn registers.

The CPU core and DMA controller can initiate bus transfers (that is, operate as bus masters) in the CPU subsystem. Modules in the CPU subsystem can only use single transfers to transfer data; burst transfer are not possible, even for the CPU core and DMA controller.

The memory area must be defined as existing within the HBUS area. For details, see *3.7.3 "Memory areas"*.

Direct access

With direct access, the CPU subsystem sends one unit of data, and that data is received as is by the HBUS module.

Buffer access

The main purpose of using buffer access is to convert a series of single transfers from the CPU subsystem into a burst transfer in the HBUS module. The buffer size is 256 bits x 2 (8 x 32 bits x 2 lines) for both reading and writing.

- Read access
  - Buffer size: 256 bits x 2 (8 × 32 bits x 2 lines)
  - The HBUS buffer fills up with data written to the HBUS module by using a burst transfer.
  - Whether to read data based on priority (critical words first) or sequentially can be selected by specifying a register setting.
  - The prefetch method can be specified (increment, decrement, or off).
- Write access
  - Buffer size: 256 bits x 2 (8 × 32 bits x 2 lines)
  - The buffer is used as a FIFO buffer.
  - Data written from the CPU subsystem as a series of single transfers is accumulated by the HBUS module and written to the access target as a burst transfer.

Writing to the buffer

The buffer uses two 8-word lines. Areas in the memory space are therefore identified in blocks of 32 bits x 8.

The block start address is aligned to an 8-word boundary, so the lower 3 bits are 000_B.

If a module in the CPU subsystem writes to the buffer area, the data is stored in the active buffer line.

Where in the buffer line the data is stored is determined by the lowest 3 bits of the address to which the data will be written.

If the data written in a subsequent write accesses matches an address used by the buffer, the data is stored in the same buffer line. If the address to which the data will be written is in a separate address block, the other buffer line will become active.

If any of the following conditions are met, the data in the buffer line will be written to the HBUS module.

- If the active line is changed (if data is written to another address block)
- If access is made in 16-bit or 8-bit units
- If data is read from an address in the block being used by the active buffer line
- If a request to flush the buffer data becomes valid

Once a buffer line becomes full of data, data in the buffer is written by the HBUS module as a burst transfer.

If the buffer line contains invalid data, the data is written by the HBUS module as a series of single transfers.

Caution When carrying out successive write accesses (up to 8), data is not written to the HBUS module until it has been confirmed that the conditions for writing the buffer line, which are coded in the software, have been satisfied.

<R>

(2) HBUS slave interface (access from the HBUS module to the CPU subsystem)

The HBUS module always uses direct access to access the CPU subsystem. Note that the HBUS bridge buffer is a 4-word (16-byte) buffer that supports burst access from the HBUS module.

3.9.2 Register overview

The HBUS bridge is controlled and operated by the following registers:

Table 3-31 Overview of HBUS bridge registers

Register name	Symbol	Address
HBUS master interface registers		
Configuration register	ETACFG	FFFF 7100$_H$
Command register	ETACMD	FFFF 7102$_H$
Wait insertion limit setting register	ETAWRL	FFFF 7106$_H$
Area 0 setting register	ETARCFG0	FFFF 7140$_H$
Area 1 setting register	ETARCFG1	FFFF 7142$_H$
Area 2 setting register	ETARCFG2	FFFF 7144$_H$
Area 3 setting register	ETARCFG3	FFFF 7146$_H$
Area 0 address register	ETARADRS0	FFFF 7150$_H$
Area 0 mask register	ETARMASK0	FFFF 7154$_H$
Area 1 address register	ETARADRS1	FFFF 7158$_H$
Area 1 mask register	ETARMASK1	FFFF 715C$_H$
Area 2 address register	ETARADRS2	FFFF 7160$_H$
Area 2 mask register	ETARMASK2	FFFF 7164$_H$
Area 3 address register	ETARADRS3	FFFF 7168$_H$
Area 3 mask register	ETARMASK3	FFFF 716C$_H$
HBUS slave interface registers		
Buffer status display register	ATEBSR	FFFF 7FDA$_H$
Address enable master ID register	ATEAPMI	FFFF 7FE4$_H$

3.9.3 Register details

(1) ETACFG – Configuration register

This register is used to specify general configuration settings for the HBUS master interface.

Access This register can be read or written in 16-bit units.

Address FFFF 7100_H

Initial value 0980_H

<R> **Caution** Be sure to clear bits 15 to 12, 10, and 6 to 0 to “0”, and set bit 7 to “1”.

<R>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	BFM	0	BPM		1	0	0	0	0	0	0	0
R	R	R	R	R/W	R	R/W	R/W	R	R	R	R	R	R	R	R

<R> **Table 3-32 ETACFG register contents**

Bit position	Bit name	Function
11	BFM	This bit specifies how the buffer is filled. 0: Sequentially 1: Based on priority (critical words first)
9, 8	BPM	These bits specify the buffer prefetch operation. 00_B: No prefetching 01_B: Prefetching occurs and the address is incremented 10_B: Prefetching occurs and the address is decremented 11_B: Reserved

(2) ETACMD – Command register

<R> This register is used to start flushing the HBUS master interface buffer.

Access This register can be read or written in 16-bit units.

Address FFFF 7102_H

Initial value 0000_H

<R> **Caution** Be sure to clear bits 15 to 12 and 0 to “0”.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<R>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BFL	0
	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W

<R> **Table 3-33 ETACMD register contents**

Bit position	Bit name	Function
1	BFL	This bit specifies whether to start flushing the buffer. Set the BFL bit to 1 to start flushing the buffer. Once the buffer has been flushed (from the HBUS bridge point of view), the BFL bit is cleared to 0. 0: Do not flush the buffer/The buffer has been flushed 1: Start flushing the buffer/The buffer is being flushed

(3) ETAWRL – HBUS wait insertion limit setting register

This register is used to specify the upper limit of the number of bus clock cycles the system will wait for until the HBUS module returns a wait response.

Caution It is strongly recommended to use this feature for debugging purposes only, and to disable the wait limit by setting this register to 0.

If a wait limit is set and the HBUS wait request exceeds the specified number of clock cycles, a wait limit error will occur.

Access This register can be read or written in 16-bit units.

Address FFFF 7106$_H$

Initial value 000F$_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	ETAWRL[7:0]							
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-34 ETAWRL register contents

Bit position	Bit name	Function
7 to 0	ETAWRL[7:0]	These bits set the upper-limit value for a wait response from HBUS. 0: No limit/Upper limit invalid Other than above: Wait for n bus clock cycles.

(4) ETARCFGn – Area n setting register

This register is used to enable the use of memory area n and specify the transfer mode for that area.

Access This register can be read or written in 16-bit units.

Address ETARCFG0: FFFF 7140_H, ETARCFG1: FFFF 7142_H

ETARCFG2: FFFF 7144_H, ETARCFG3: FFFF 7146_H

Initial value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	MODE[3:0]				0	0	0	EN
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R	R	R	R/W

Table 3-35 ETARCFGn register contents

Bit position	Bit name	Function
<R> 7 to 4	MODE[3:0]	These bits specify the transfer mode for area n. 0000_B: Direct access 0001_B: Access via buffer Other than the above: Setting prohibited
0	EN	Area n enable bit 0: Disable 1: Enable

Note <R> If memory spaces overlap in the specified area, the transfer mode is specified by priority as follows: Access via buffer > Direct access

(5) ETARADRSn – Area n address register

This register is used to specify the base address that specifies memory area n.

Access This register can be read or written in 32-bit units.

Address ETARADRS0: FFFF 7150_H, ETARADRS1: FFFF 7158_H

ETARADRS2: FFFF 7160_H, ETARADRS3: FFFF 7168_H

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETARADRS[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETARADRS[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-36 ETARADRSn register contents

Bit position	Bit name	Function
31 to 0	ETAR ADRS[31:0]	These bits specify the base address that specifies memory area n. Bit 28 is the sign bit for bits 31 to 29, which store the sign-extended value of the address. Be sure to clear bits 11 to 0 to “0”.

(6) ETARMASKn – Area n mask register

This register is used to mask the base address that specifies memory areas 0 to 3. Specify 1s in succession, starting from the lower bits.

Access This register can be read or written in 32-bit units.

Address ETARMASK0: FFFF 7154_H, ETARMASK1: FFFF $715C_H$

ETARMASK2: FFFF 7164_H, ETARMASK3: FFFF $716C_H$

Initial value $1FFF\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETARMASK[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETARMASK[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

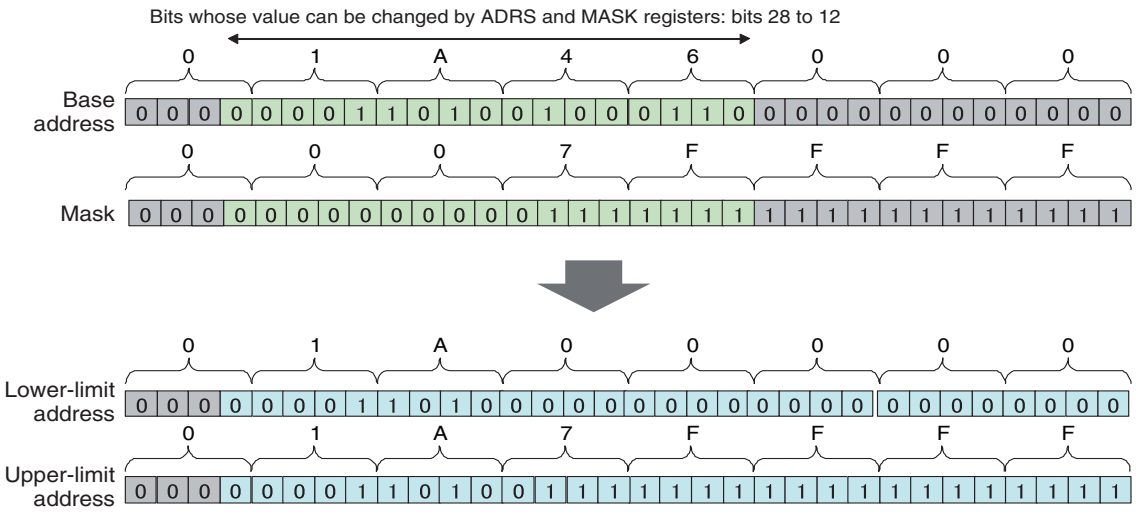
Table 3-37 ETARMASKn register contents

Bit position	Bit name	Function
31 to 0	ETAR MASK[31:0]	These bits mask the base address that specifies memory area n. Be sure to clear bits 31 to 29 to “0”. Be sure to set bits 11 to 0 to “1”.

(7) Calculating the address range for area n

Address areas are specified by using base addresses and by masking. The top and bottom addresses of an area can be manipulated by masking the base address. The bottom address is the address whose bits are set to 1 but cleared to 0 in the masked address, and the top address is the address whose bits are set to 1 and remain 1 in the masked address. See the figure below for a setting example.

Caution To set the mask value, set 1s in succession from the lower bits.



Example: If 01A4 6000_H is set as the base address and 0007 $FFFF_H$ is set as the mask:
area from 01A0 0000_H to 01A7 $FFFF_H$ is specified.
Upper-limit address: 01A0 0000_H
Lower-limit address: 01A7 $FFFF_H$

Figure 3-15 Example of calculating the address range for area n

(8) ATEBSR – Buffer status display register

This register indicates the status of the HBUS slave interface's buffer. This buffer stores the data to be sent from the HBUS module as a burst access. If the buffer is empty, it means that the HBUS is not accessing the CPU subsystem.

Access This register can be read in 16-bit units.

Address FFFF 7FDA$_H$

Initial value 0001$_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	E
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 3-38 ATEBSR register contents

Bit position	Bit name	Function
0	E	This bit indicates the transfer buffer status. 0: There is data in the transfer buffer, and the data is being transferred. 1: There is no data in the transfer buffer because transfer is complete.

(9) ATEAPMI – Address enable master ID register

This register enables a master connected to the HBUS slave interface to access the CPU subsystem.

Access This register can be read or written in 16-bit units.

Address FFFF 7FE4$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EN0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W

Table 3-39 ATEAPMI register contents

Bit position	Bit name	Function
0	EN0	This bit enables an HBUS master to access the CPU subsystem. 0: Access prohibited 1: Access enabled

3.10 Write-Protected Registers

Write-protected registers are protected from being accessed by mistake such as by an erroneous program command.

Write-protected registers can only be written by using a special protection unlock sequence.

3.10.1 Register protection clusters

Protected registers are grouped into several register protection clusters.

The protection mechanism treats all registers in the same cluster as a single protection unit.

Once a protection unlock sequence is initiated for a register, the other registers in that protection cluster cannot be accessed. Otherwise, the unlock sequence would be disrupted and writing to the register would fail.

The figure below shows a disruption to the unlock sequence by an access to a register in the same cluster within an interrupt service routine.

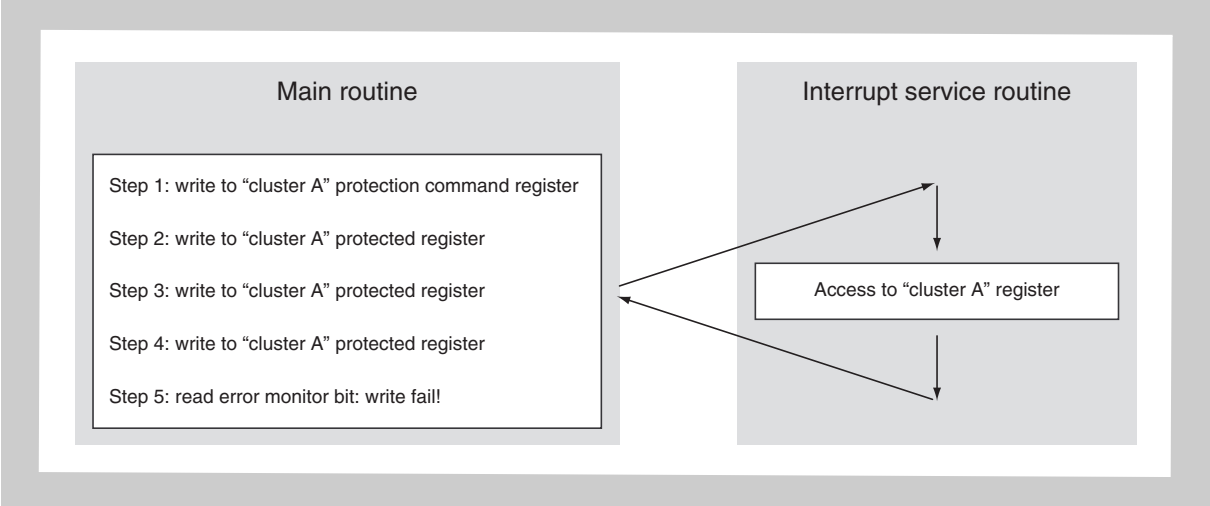


Figure 3-16 Disruption to register protection unlock sequence

If a register in another protection cluster is accessed while an unlock sequence is being executed, the unlock sequence remains unaffected and the register can be successfully written to.

An example of a successfully completed protection unlock sequence is shown below.

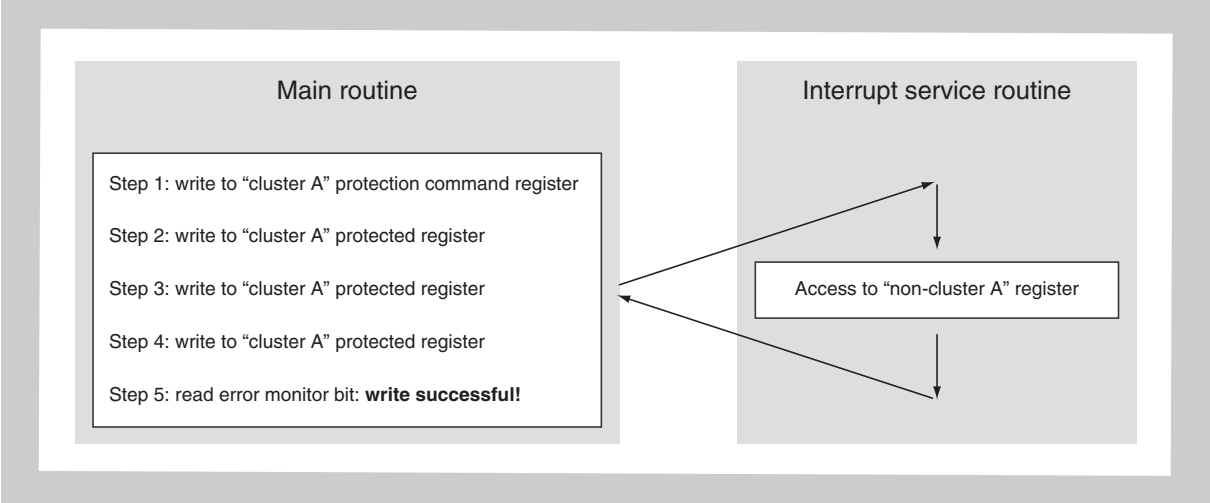


Figure 3-17 Example of successful completion of register protection unlock sequence

For how to write to write-protected registers in the V850E2/Sx4-H, see *3.10.4 “Write-protected registers in the V850E2/Sx4-H”*.

3.10.2 Register protection unlock sequence

Write-protected registers can only be written by using a special protection unlock sequence.

1. Write $A5_H$ to the protection command register.
2. Write the desired value to the protected register.
3. Write the bit-wise inversion of the desired value to the protected register.
4. Write the desired value to the protected register.
5. Confirm that the desired value has been successfully written to the protected register by verifying that the error monitor bit in the protection status register is “0”. If the error monitor bit is 1, start again from step 1.

If another register is accessed between steps 1 and 4, the protection mechanism behaves as follows:

- If the second register belongs to the same cluster, the write to the protected register fails (the error monitor bit indicates 1). The entire sequence has to be re-executed from step 1.
- If the second register does not belong to the same cluster, writing to the protected register is not interrupted, and is completed normally.

3.10.3 Register protection and interrupt/emulation breaks

If the protection unlock sequence is interrupted, the protection mechanism behaves as follows:

(1) Interrupt during protection unlock sequence

If an interrupt is acknowledged during the above protection unlock sequence and the interrupt service routine does not access a register in the same register protection cluster, the protection unlock sequence is not disrupted and the write to the protected register can be successfully completed after returning from the interrupt service routine.

(2) Emulation break during protection unlock sequence

If an emulation break occurs during the above protection sequence, for instance because of a breakpoint hit, register protection is suspended until normal operation is resumed after the break.

This means that, even if a register in the same cluster is accessed during a break, the protected sequence is not disrupted and the error monitor bit is not set to 1.

3.10.4 Write-protected registers in the V850E2/Sx4-H

This section describes the write-protected registers in the V850E2/Sx4-H.

Table 3-40 Write-protected registers

Module	Protected registers	Protection enable registers		Protection cluster
		Command register	Status register	
Clock controller	CKSC_0n	PROTCMD0	PROTS0	Control protection cluster 0
Clock controller	CKSC_1n	PROTCMD1	PROTS1	Control protection cluster 1
Clock controller	PLLEk	PROTCMD2	PROTS2	Control protection cluster 2
	MOSCE			
	SOSCE			
	ROSCE			
	CKSC_An			
Standby circuit	PSC0			
	PSC1			
Reset circuit	SWRESA			
	LVICNT			
On-chip debugger	IDMODI	PROTCMD3	PROTS3	Control protection cluster 3
Clock monitor	CLMAnCTL0	CLMAPCMD	CLMAnPS	Clock monitor protection cluster
Ports[a]	PDSCn, JPDSCnPODCn, JPODCn	PPCMDn	PPROTSn	Port protection clusters 1 to 4
Self programming feature	FLMDCNT	FLMDPCMD	FLMDPS	Self programming protection cluster

a) Each port group has its own protection command register and protection status register. For details, see the description about port control register protection in *(1) "Port protection clusters" on page 203*.

(1) Port protection clusters

The following port registers are write-protected:

- Port drive strength control registers (PDSCn and JPDSCn)
- Port open drain control registers (PODCn and JPODCn)

The above port control registers, which are assigned to the port group indicated by n, are allocated to four port protection clusters.

Table 3-41 Port protection clusters

Port protection cluster	Port group
1	JP0
2	P0
3	P1, P3, P10
4	P21, P24, P25, P26 to P28

Note Each port group (indicated by n) has its own port protection command register (PPCMDn) and port protection status register (PPROTSn).

Register size The protected port control registers and corresponding protection command registers (PPCMDn) are 8-bit registers.

Caution The port protection command registers (PPCMDn) and all the protected registers must be accessed in 8-bit units.

The protection unlock sequence is shown below.

1. Write $A5_H$ to the protection command register (PPCMDn).
2. Write a 32-bit value to the protected register that is the desired value ($xxxx_H$) with the upper 16 bits [31:16] set to 0 (0000 $xxxx_H$).
3. Write the bit-wise inversion of the desired value to the protected register so that the upper 16 bits [31:16] are set to FFFF $xxxx_H$.
4. Write a 32-bit value to the protected register that is the desired value ($xxxx_H$) with the upper 16 bits [31:16] set to 0 (0000 $xxxx_H$).
5. Confirm that the desired value has been successfully written to the protected register by verifying that the PPROTSn.PPROTSnERR bit is 0. If this bit is 1, start again from step 1.

3.10.5 Overview of protected registers in the V850E2/Sx4-H

Register write protection is controlled and operated by the following registers:

Table 3-42 Protection command registers (1/2)

Register name	Symbol	Address
Control protection cluster		
Protection command register 0	PROTCMD0	FF42 4000$_H$
Protection command register 1	PROTCMD1	FF42 8000$_H$
Protection command register 2	PROTCMD2	FF42 0300$_H$
Protection command register 3	PROTCMD3	FF42 0308$_H$
Protection status register 0	PROTS0	FF42 4004$_H$
Protection status register 1	PROTS1	FF42 8004$_H$
Protection status register 2	PROTS2	FF42 0304$_H$
Protection status register 3	PROTS3	FF42 030C$_H$
Clock monitor cluster		
CLMA0:		
Protection command register	CLMA0PCMD	FF80 2010$_H$
Protection status register	CLMA0PS	FF80 2014$_H$
CLMA2:		
Protection command register	CLMA2PCMD	FF80 4010$_H$
Protection status register	CLMA2PS	FF80 4014$_H$
CLMA3:		
Protection command register	CLMA3PCMD	FF80 5010$_H$
Protection status register	CLMA3PS	FF80 5014$_H$
Port protection clusters		
Port protection cluster 1		
Protection command register	JPPCMD0	FF44 04C0$_H$
Protection status register	JPPROTS0	FF44 04B0$_H$
Port protection cluster 2		
Protection command register	PPCMD0	FF40 4C00$_H$
Protection status register	PPROTS0	FF40 4B00$_H$
Port protection cluster 3		
Protection command registers	PPCMD1	FF40 4C04$_H$
	PPCMD3	FF40 4C0C$_H$
	PPCMD10	FF40 4C28$_H$
Protection status registers	PPROTS1	FF40 4B04$_H$
	PPROTS3	FF40 4B0C$_H$
	PPROTS10	FF40 4B28$_H$

Table 3-42 Protection command registers (2/2)

Register name	Symbol	Address
Port protection cluster 4		
Protection command registers	PPCMD21	FF40 4C54$_H$
	PPCMD24	FF40 4C60$_H$
	PPCMD25	FF40 4C64$_H$
	PPCMD26	FF40 4C68$_H$
	PPCMD27	FF40 4C6C$_H$
	PPCMD28	FF40 4C70$_H$
Protection status registers	PPROTS21	FF40 4B54$_H$
	PPROTS24	FF40 4B60$_H$
	PPROTS25	FF40 4B64$_H$
	PPROTS26	FF40 4B68$_H$
	PPROTS27	FF40 4B6C$_H$
	PPROTS28	FF40 4B70$_H$
Self programming protection cluster		
FLMD protection command register	FLMDPCMD	FF43 8004$_H$
FLMD protection error status register	FLMDPS	FF43 8008$_H$

3.10.6 Detailed description of control protection clusters and registers

(1) PROTCMDn – Protection command register

This register is used to initiate the write protection unlock sequence.

Index n n indicates the number of the protection command register. For details, see *Table 3-42 "Protection command registers"*.

Access This register is write-only, in 8-bit units. This bit is always read as 00_H.

Address For details, see *Table 3-42 "Protection command registers"*.

Initial value 00_H. This register is initialized by any reset.

<R>

7	6	5	4	3	2	1	0
PCMD[7:0]							
W	W	W	W	W	W	W	W

The usage of the PROTCMDn register is described in *3.10.2 "Register protection unlock sequence"*.

Table 3-43 PROTCMDn register contents

Bit position	Bit name	Function
<R> 7 to 0	PCMD[7:0]	These bits indicate the protection command to enable writing to Isolated area m registers.

(2) PROTSn – Protection status register

This register shows the status of the protection unlock sequence initiated by PROTCMDm.

Index n n indicates the number of the protection command register. For details, see *Table 3-42 "Protection command registers"*.

Access This register is read-only, in 8-bit units.

Write operation is ignored.

Address For details, see *Table 3-42 "Protection command registers"*.

Initial value 00_H. This register is initialized when a reset occurs, or A5H is written to the PROTCMDn register.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PROTERR
R	R	R	R	R	R	R	R

Table 3-44 PROTSn register contents

Bit position	Bit name	Function
0	PROTERR	This bit indicates a write sequence protection error. 0: No protection error 1: Protection error occurred

3.10.7 Detailed description of clock monitor protection clusters and registers

(1) CLMAnPCMD – CLMAn protection command register

This is the protection command register for the CLMAnCTL0 register.

Index n n indicates the number of the protection command register. For details, see *Table 3-42 "Protection command registers"*.

Access This register is write-only, in 8-bit units.

Address For details, see *Table 3-42 "Protection command registers"*.

Initial value Undefined

7	6	5	4	3	2	1	0
CLMAnREG[7:0]							
W	W	W	W	W	W	W	W

Table 3-45 CLMAnPCMD register contents

Bit position	Bit name	Function
7 to 0	CLMAnREG[7:0]	These bits indicate the protection command to enable writing to the CLMAnCTL0 register.

(2) CLMAnPS – CLMAn protection status register

This register is used to check whether or not writing to a write-protected register (CLMAnCTL0) has been executed successfully.

Index n n indicates the number of the protection command register. For details, see *Table 3-42 "Protection command registers"*.

Access This register is read-only, in 8-bit units.

Address For details, see *Table 3-42 "Protection command registers"*.

Initial value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	CLMAnPRERR
R	R	R	R	R	R	R	R

Table 3-46 CLMAnPS register contents

Bit position	Bit name	Function
0	CLMAnPRERR	This bit is used to check whether or not writing to a write-protected register (CLMAnCTL0) has been executed successfully. 0: Write operation successful 1: Write operation failed

3.10.8 Detailed description of port protection clusters and registers

(1) PPCMDn – Port protection command register

The PPCMDn register is the protection command register for port group n.

Index n n indicates the number of the protection command register. For details, see *Table 3-42 "Protection command registers"*.

Access This register is write-only, in 8-bit units.

This bit is always read as 00000000H.

Address For details, see *Table 3-42 "Protection command registers"*.

Initial value 00000000_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W

Table 3-47 PPCMDn register contents

Bit position	Bit name	Function
7 to 0	–	These bits indicate the protection command to enable writing to a port register.

(2) PPROTSn – Port protection status register

The PPROTSn register indicates the protection status of the write-protected registers in port group n. This register shows the status of the protection sequence initiated by PPCMDn.

Index n n indicates the number of the protection command register. For details, see *Table 3-42 "Protection command registers"*.

Access This register is read-only, in 8-bit units.

Write operation is ignored.

Address For details, see *Table 3-42 "Protection command registers"*.

Initial value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PPROTSnPRERR
R	R	R	R	R	R	R	R

Table 3-48 PPROTSn register contents

Bit position	Bit name	Function
0	PPROTSn PRERR	This bit indicates a write sequence protection error. 0: No protection error 1: Protection error occurred

3.10.9 Detailed description of self programming protection clusters and registers

(1) FLMDPCMD – FLMD protection command register

This is the protection command register for the FLMDCNT register.

Access This register is write-only, in 8-bit units.

Address For details, see *Table 3-42 "Protection command registers"*.

Initial value

7	6	5	4	3	2	1	0
FLMDPC[7:0]							
W	W	W	W	W	W	W	W

Table 3-49 FLMDPCMD register contents

Bit position	Bit name	Function
7 to 0	FLMDPC[7:0]	These bits indicate the protection command to enable writing to the FLMDCNT register.

(2) FLMDPS – FLMD protection status register

This bit is used to check whether or not writing to a write-protected register (FLMDCNT) has been executed successfully.

Access This register is read-only, in 8-bit units.

Address For details, see *Table 3-42 "Protection command registers"*.

Initial value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FLMDPRERR
R	R	R	R	R	R	R	R

Table 3-50 FLMDPS register contents

Bit position	Bit name	Function
0	FLMDPRERR	This bit is used to check whether or not writing to a write-protected register (FLMDCNT) has been executed successfully. 0: Write operation successful 1: Write operation failed

<R>

3.11 System Error Report Configuration Registers

(1) SEG_CONT – SYSERR exception report configuration register

This register enables/disables reporting of SYSERR exceptions caused by system errors.

Setting a bit of this register to 1 enables reporting of SYSERR when the corresponding error occurs.

Clearing a bit of this register to 0 disables reporting of SYSERR when the corresponding error occurs, although the corresponding error flag will be set.

Be sure to clear bits 15 to 8, 6, 4, 3, and 0 to “0”.

Access This register can be read or written in 16-, 8-, or 1-bit units.

Address FFFF 64B0$_H$

Initial Value 0000$_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
DMAE	0	SEGE	0	0	EXTE	FCHE	0
R/W	R	R/W	R	R	R/W	R/W	R

Table 3-51 SEG_CONT register contents

Bit position	Bit name	Function
7	DMAE	DMA error report enable Enables reporting of SYSERR when a DMA error occurs.
5	SEGE	Reserved area access report enable Enables reporting of SYSERR when the CPU accesses data in a reserved area.
2	EXTE	EXT area error report enable Enables reporting of SYSERR when the CPU accesses data in an external memory or a peripheral I/O area.
1	FCHE	Code flash error report enable Enables reporting of SYSERR when the flash memory outputs an error indicating that the CPU has accessed data in the code flash area.

(2) SEG_FLAG – System error trigger save register

This register contains flags that save the error status when a SYSERR exception occurs.

When a SYSERR exception occurs, the corresponding error trigger flag is set to 1.

After a flag that is 1 is read, it can be cleared by writing 0. If a flag is 0 when it is read, even if 0 is subsequently written, the flag will be set to 1 if an error occurs between when the flag is read and written.

Be sure to set bits 15 to 8, 6, 4, 3, and 0 to “0”.

Access This register can be read or written in 16-, 8-, or 1-bit units.

Address FFFF 64B2$_H$

Initial Value 0000$_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
DMAF	0	SEGF	0	0	EXTF	FCHF	0
R/W	R	R/W	R	R	R/W	R/W	R

Table 3-52 WDTAnEVAC register contents

Bit position	Bit name	Function
7	DMAF	DMA error flag This flag is set if a system error caused by DMA access occurs.
5	SEGF	Reserved area error flag This flag is set if the CPU accesses data in a reserved area.
2	EXTF	EXT area error flag This flag is set if an error occurs because the CPU has accessed data in an external memory or a peripheral I/O area.
1	FCHF	Code flash error flag This flag is set if the flash memory outputs an error indicating that the CPU has accessed data in the code flash area.

Chapter 4 External Memory Controller (MEMC)

4.1 Features

The external memory access controller (MEMC) of the V850E2/Sx4-H provides an interface for connecting external memory, ASICs, etc., to this microcontroller. The V850E2/Sx4-H allows the connection of the following two types of memory:

- SRAM
- SDRAM

The V850E2/Sx4-H provides four chip select areas (CS2 to CS4 and CS7), and the bus size and wait time can be specified independently for each chip select area.

This chapter describes how to specify the settings for eight chip select areas, but the initial value can only be changed for four areas (CS2 to CS4 and CS7) for the V850E2/Sx4-H.

I/O signals The I/O signals of the MEMC are shown in the following table:

Table 4-1 I/O signals of MEMC (1/2)

MEMC signal	Function	Connected to:
MAOxx	Bus for outputting addresses to external memory in the separate bus mode	Port MEMC0A[0:25]
MDIxx	Bus for inputting data to external memory in the separate bus mode	Ports MEMC0AD[0:27] and MEMC0D[28:31]
MADxx	Bus for outputting data to external memory in the separate bus mode and outputting address/data in the multiplexed bus mode	Ports MEMC0AD[0:27] and MEMC0D[28:31]
BENZxx	Byte enable signal output (D0 to D7)	Port $\overline{\text{MEMC0BEN0}}$
	Byte enable signal output (D8 to D15)	Port $\overline{\text{MEMC0BEN1}}$
	Byte enable signal output (D16 to D23)	Port $\overline{\text{MEMC0BEN2}}$
	Byte enable signal output (D24 to D31)	Port $\overline{\text{MEMC0BEN3}}$
RDZ	Read strobe signal output	Port $\overline{\text{MEMC0RD}}$
DVCLK	Bus clock output	Port MEMC0CLK
WRZ	Write strobe signal output	Port $\overline{\text{MEMC0WR}}$
ASTBZ	Address strobe signal output for external memory	Port $\overline{\text{MEMC0ASTB}}$
DSTBZ	Latch strobe signal output on external address bus	Port $\overline{\text{MEMC0DSTB}}$

Table 4-1 I/O signals of MEMC (2/2)

MEMC signal	Function	Connected to:
DQMxx	I/O mask signal output for SDRAM (D0 to D7)	Port MEMC0DQM0
	I/O mask signal output for SDRAM (D8 to D15)	Port MEMC0DQM1
	I/O mask signal output for SDRAM (D16 to D23)	Port MEMC0DQM2
	I/O mask signal output for SDRAM (D24 to D31)	Port MEMC0DQM3
SDRASZ	Row address strobe signal output for SDRAM	Port $\overline{\text{MEMC0SDRAS}}$
SDCASZ	Column address strobe signal output for SDRAM	Port $\overline{\text{MEMC0SDCAS}}$
CKE	SDRAM clock enable output signal	Port MEMC0CKE
SDWEZ	Data write enable output for SDRAM	Port $\overline{\text{MEMC0WE}}$
MAOxx	Address bus for external memory	Port MEMC0A[0:25]
CSZxx	Chip select signal output	Ports $\overline{\text{MEMC0CS}}$[2:4] and $\overline{\text{MEMC0CS7}}$
WAITZ	External wait request input	Port $\overline{\text{MEMC0WAIT}}$
HLDAKZ	Bus hold acknowledge output	Port $\overline{\text{MEMC0HLDAK}}$
HLDRQZ	Bus hold request input	Port $\overline{\text{MEMC0HLDRQ}}$

4.1.1 Operation modes and connectable memory types

The separate bus mode and multiplexed bus mode are available as the operation modes.

Mode settings can be specified by using flash mask option bit OPBT0[27].

(1) Separate bus mode

This is an operation mode that connects address output and data input/output to external memory using independent signal lines. By using this mode, either of the following two types of memory can be connected to individual chip select areas.

- SRAM
- SDRAM (SDRAM can only be selected for a specific chip select area (CS4).)

(2) Multiplexed bus mode

This is an operation mode that connects address output and data input/output to external memory by using the same signal lines. In this mode, the memory that can be connected is limited to SRAM for all chip select areas, but it is possible to reduce the number of pins required for external memory connection.

4.1.2 External bus clock

The CKSCLK_000 clock divided by 2 or 4 can be selected for the external bus clock.

The flash mask option bits OPBT0.OPBT0[2:1] are used to select the bus clock.

4.1.3 Chip select signal output

The external bus area in the memory space is divided into four chip select areas, and a chip select signal can be output for each chip select area. The allocation of these chip select areas is fixed by the system and cannot be changed through programming.

4.1.4 Operation setting function

Operation can be enabled or disabled by using the BCT0 and BCT1 registers for each chip select area.

4.1.5 Bus sizing function

The bus size can be selected from 8 bits, 16 bits, or 32 bits for each chip select area. If accessing memory of a size greater than the selected bus width is attempted, the bus sizing feature divides the data into sizes smaller than the bus width.

4.1.6 Data endianness setting function

The data endianness (little-endian or big-endian) can be specified for the chip select areas. However, because the software development tools made by Renesas Electronics, such as assemblers and debuggers, only support little-endian, instruction fetching in big-endian is not possible.

Little-endian is specified for all chip select areas in the initial state.

When big-endian is selected, accessing a misaligned address in these areas is prohibited.

4.1.7 Programmable wait setting functions

This microcontroller has the following wait setting features, which can be specified for each chip select area.

- Programmable data wait
- Data hold wait
- Data setup wait (in multiplexed bus mode, during write access)
- Address setup wait (in multiplexed bus mode)
- Address hold wait (in multiplexed bus mode)
- Idle cycles
- RAS latency (when accessing SDRAM)
- CAS latency (when accessing SDRAM)

4.1.8 External wait function

Data wait cycles of any width can be externally inserted by using the WAITZ pin during write access to SRAM. The WAITZ pin signal is sampled just before the data output cycle, and the data latch timing can be delayed by any amount.

4.1.9 External wait error detection function

If an external wait cycle is continuously input for 128 clock cycles due to a defect or other problem at the external wait pin, the wait state is canceled and a SYSERROR interrupt occurs to prevent a system hang-up.

4.1.10 Bus hold function

A bus hold request from outside can be input to the HLDRQZ pin.

4.2 Registers

These registers are used to control the MEMC registers.

Note The clock counts shown in *4.2 "Registers"* indicate the clock count when operating on the external bus clock unless otherwise specified.

Table 4-2 External memory access control registers

Address	Register name	Symbol	R/W	Access bit unit			Initial value
				1	8	16	
$FFFF7200_H$	Bus size configuration register	BSC	R/W			√	a
$FFFF7202_H$	Data endian configuration register	DEC	R/W			√	0000_H
$FFFF7204_H$	Bus cycle type configuration register 0	BCT0	R/W			√	a
$FFFF7206_H$	Bus cycle type configuration register 1	BCT1	R/W			√	a
$FFFF7208_H$	Data wait configuration register 0	DWC0	R/W			√	a
$FFFF720A_H$	Data wait configuration register 1	DWC1	R/W			√	a
$FFFF720C_H$	Data hold wait configuration register	DHC	R/W			√	0000_H
$FFFF720E_H$	Data setup wait configuration register	DSC	R/W			√	0000_H
$FFFF7210_H$	Address wait configuration register 0	AWC0	R/W			√	a
$FFFF7212_H$	Address wait configuration register 1	AWC1	R/W			√	a
$FFFF7214_H$	Address cycle configuration register 0	ICC0	R/W			√	0000_H
$FFFF7216_H$	Address cycle configuration register 1	ICC1	R/W			√	0000_H
$FFFF721A_H$	External wait error configuration register	EWC	R/W			√	0000_H
$FFFF7220_H$	SDRAM enable control register	SEN	R/W		√		00_H
$FFFF7222_H$	SDRAM configuration register	SDCR	R/W			√	$20C0_H$
$FFFF7224_H$	SDRAM status register	STR	R		√		00_H
$FFFF7226_H$	SDRAM refresh control register	RFS	R/W			√	0000_H

a) Depends on the product

4.2.1 BSC - Bus size configuration register

The BSC register is used to specify the external bus size for each chip select area.

Access This register can be read or written in 16-bit units.

Address $FFFF7200_H$

Initial value Depends on the product

Caution Only bits 15, 14, and 9 to 4 can be set.

15	14	13	12	11	10	9	8
BS71	BS70	BS61	BS60	BS51	BS50	BS41	BS40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
BS31	BS30	BS21	BS20	BS11	BS10	BS01	BS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-3 BSC register contents

Bit position	Bit name	Function
15:14, 13:12, 11:10, 9:8, 7:6, 5:4, 3:2, 1:0	BSn1, BSn0	Bus size configuration bit Specify the bus width of each chip select area.

BSn1	BSn0	Bus size
0	0	8 bits
0	1	16 bits
1	0	32 bits
1	1	Setting prohibited

4.2.2 DEC - Data endian configuration register

The DEC register is used to specify the endianness of the external bus.

Access This register can be read or written in 16-bit units.

Address $FFFF7202_H$

Initial value 0000_H. This register is initialized by any reset.

Cautions
1. Only bits 14, 8, 6, and 4 can be set. Be sure to clear bits 15, 13 to 9, 7, 5, and 3 to 0 to “0”.
2. Accessing a misaligned address in these areas is prohibited when big-endian is selected.

15	14	13	12	11	10	9	8
0	DE7	0	DE6	0	DE5	0	DE4
R	R/W	R	R/W	R	R/W	R	R/W
7	6	5	4	3	2	1	0
0	DE3	0	DE2	0	DE1	0	DE0
R	R/W	R	R/W	R	R/W	R	R/W

Table 4-4 DEC register contents

Bit position	Bit name	Function
14, 12, 10, 8, 6, 4, 2, 0	DEn	Data endian configuration bits Specify the endianness of each chip select area. 0: Little-endian 1: Big-endian

4.2.3 BCT0, BCT1 - Bus cycle type configuration registers 0 and 1

The BCT0 and BCT1 registers are used to specify a bus cycle type on the external bus, for each chip select area.

Access This register can be read or written in 16-bit units.

Address $FFFF7204_H$: BCT0

$FFFF7206_H$: BCT1

Initial value Depends on the product

Cautions

1. Only bits 15, 13 to 11, 9, 8, 3, 1, and 0 can be set.
2. Specify the settings for the BCT0 and BCT1 registers immediately after a reset ends and do not change their set values. In particular, do not change their values during a burst read. The operation is not guaranteed if the values are changed.

BCT0

15	14	13	12	11	10	9	8
ME3	0	BCT31	BCT30	ME2	0	BCT21	BCT20
a	R	R/W	R/W	a	R	R/W	R/W
7	6	5	4	3	2	1	0
ME1	0	BCT11	BCT10	ME0	0	BCT01	BCT00
a	R	R/W	R/W	a	R	R/W	R/W

BCT1

15	14	13	12	11	10	9	8
ME7	0	BCT71	BCT70	ME6	0	BCT61	BCT60
a	R	R/W	R/W	a	R	R/W	R/W
7	6	5	4	3	2	1	0
ME5	0	BCT51	BCT50	ME4	0	BCT41	BCT40
a	R	R/W	R/W	a	R	R/W	R/W

a) Attributes vary depending on the product.

Table 4-5 BCT0 and BCT1 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>13:12,
9:8,
5:4,
1:0</td><td>BCTn1,
BCTn0</td><td>Bus size configuration bits
Specify the memory (bus cycle type) to be connected to each chip select area in the separate bus mode.
In the multiplexed bus mode, the values specified for the BCTn1 and BCTn0 bits are ignored, and SRAM type bus cycles are always generated (n = 0 to 7).
The BCT41 and BCT40 bit settings do not affect SDRAM controller operations.
Only SDRAM can be connected to the CS4 area.
<table>
<tr><th>BCTn1</th><th>BCTn0</th><th>Memory type</th></tr>
<tr><td>0</td><td>0</td><td>SRAM</td></tr>
<tr><td colspan="2">Other than above</td><td>Setting prohibited</td></tr>
</table>
</td></tr>
<tr><td>15, 11,
7, 3</td><td>MEn</td><td>Memory controller enable bit
Specify whether to enable or disable the microcontroller’s on-chip MEMC for each chip select area.
0: Memory controller stopped (external bus cycles are not generated)
1: Memory controller enabled
If the MEMC is disabled, the microcontroller does not generate an external bus cycle, and the current reading/writing finishes (n = 0 to 7).</td></tr>
</table>

4.2.4 DWC0, DWC1 - Data wait configuration registers 0 and 1

The DWC0 and DWC1 registers are used to specify the number of data wait cycles of the external bus.

The values specified for the DWC0 and DWC1 registers are valid for the following bus cycles:

- SRAM bus type data transfer cycles
- Data transfer cycles in the multiplexed bus mode

Access This register can be read or written in 16-bit units.

Address $FFFF7208_H$: DWC0

$FFFF720A_H$: DWC1

Initial value Depends on the product

Caution For the DWC0 register, only bits 15 to 8 can be set. For the DWC1 register, only bits 15 to 12 and 3 to 0 can be set.

DWC0

15	14	13	12	11	10	9	8
DW33	DW32	DW31	DW30	DW23	DW22	DW21	DW20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DW13	DW12	DW11	DW10	DW03	DW02	DW01	DW00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DWC1

15	14	13	12	11	10	9	8
DW73	DW72	DW71	DW70	DW63	DW62	DW61	DW60
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DW53	DW52	DW51	DW50	DW43	DW42	DW41	DW40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-6 DWC0 and DWC1 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15 to 12,
11 to 8,
7 to 4,
3 to 0</td><td>DWn3,
DWn2,
DWn1,
DWn0</td><td>Data wait configuration bits
Specify the number of data wait cycles for each chip select area.
<table>
<tr><th>DWn3</th><th>DWn2</th><th>DWn1</th><th>DWn0</th><th>Number of data wait cycles</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>No data wait cycles</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1 clock cycle</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>2 clock cycles</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>3 clock cycles</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>4 clock cycles</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>5 clock cycles</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>6 clock cycles</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>7 clock cycles</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>8 clock cycles</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>9 clock cycles</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>10 clock cycles</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>11 clock cycles</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>12 clock cycles</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>13 clock cycles</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>14 clock cycles</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>15 clock cycles</td></tr>
</table></td></tr>
</table>

4.2.5 DHC - Data hold wait configuration register

The DHC register is used to specify the number of extended data hold wait cycles for each chip select area in the write cycle on the external bus.

The number of data hold wait cycles determined by the DHC register value + one cycle is inserted in a write cycle.

Access This register can be read or written in 16-bit units.

Address $FFFF720C_H$

Initial value 0000_H. This register is initialized by any reset.

Caution Only bits 15, 14, and 9 to 4 can be set. Be sure to clear bits 13 to 10 and 3 to 0 to “0”.

15	14	13	12	11	10	9	8
DH71	DH70	DH61	DH60	DH51	DH50	DH41	DH40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DH31	DH30	DH21	DH20	DH11	DH10	DH01	DH00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-7 DHC register contents

Bit position	Bit name	Function
15:14, 13:12, 11:10, 9:8, 7:6, 5:4, 3:2, 1:0	DHn1, DHn0	Data hold wait configuration bits Specify the number of data hold wait cycles for each chip select area.

DHn1	DHn0	Number of data hold wait cycles
0	0	No data hold wait cycles
0	1	1 clock cycle
1	0	2 clock cycles
1	1	3 clock cycles

4.2.6 DSC - Data setup wait configuration register

The DSC register is used to specify the number of data setup wait cycles on the external bus in the multiplexed bus mode for each chip select area.

Access This register can be read or written in 16-bit units.

Address $FFFF720E_H$

Initial value 0000_H. This register is initialized by any reset.

Caution Only bits 15, 14, and 9 to 4 can be set. Be sure to clear bits 13 to 10 and 3 to 0 to “0”.

15	14	13	12	11	10	9	8
DS71	DS70	DS61	DS60	DS51	DS50	DS41	DS40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DS31	DS30	DS21	DS20	DS11	DS10	DS01	DS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-8 DSC register contents

Bit position	Bit name	Function
15:14, 13:12, 11:10, 9:8, 7:6, 5:4, 3:2, 1:0	DSn1, DSn0	Data setup wait configuration bits Specify the number of data setup wait cycles for each chip select area.

DSn1	DSn0	Number of data setup wait cycles
0	0	No data setup wait cycles
0	1	1 clock cycle
1	0	2 clock cycles
1	1	3 clock cycles

4.2.7 AWC0, AWC1 - Address wait configuration registers 0 and 1

The AWC registers are used to specify the address wait period on the external bus for each chip select area.

The values specified for the AWC registers are valid only in the multiplexed bus mode.

Access This register can be read or written in 16-bit units.

Address $FFFF7210_H$: AWC0

$FFFF7212_H$: AWC1

Initial value Depends on the product

Caution For the AWC0 register, only bits 15 to 8 can be set. For the AWC1 register, only bits 15 to 12 and 3 to 0 can be set.

AWC0

15	14	13	12	11	10	9	8
AHW31	AHW30	ASW31	ASW30	AHW21	AHW20	ASW21	ASW20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
AHW11	AHW10	ASW11	ASW10	AHW01	AHW00	ASW01	ASW00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

AWC1

15	14	13	12	11	10	9	8
AHW71	AHW70	ASW71	ASW70	AHW61	AHW60	ASW61	ASW60
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
AHW51	AHW50	ASW51	ASW50	AHW41	AHW40	ASW41	ASW40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-9 AWC0 and AWC1 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>13:12,
9:8,
5:4,
1:0</td><td>ASWn1,
ASWn0</td><td>Address setup wait configuration bits
Specify the number of address setup wait cycles for each chip select area.
<table>
<tr><th>ASWn1</th><th>ASWn0</th><th>Number of address setup wait cycles</th></tr>
<tr><td>0</td><td>0</td><td>No address setup wait cycles</td></tr>
<tr><td>0</td><td>1</td><td>1 clock cycle</td></tr>
<tr><td>1</td><td>0</td><td>2 clock cycles</td></tr>
<tr><td>1</td><td>1</td><td>3 clock cycles</td></tr>
</table></td></tr>
<tr><td>15:14,
11:10,
7:6,
3:2</td><td>AHWn1,
AHWn0</td><td>Address hold wait configuration bits
Specify the number of address hold wait cycles for each chip select area.
<table>
<tr><th>AHWn1</th><th>AHWn0</th><th>Number of address hold wait cycles</th></tr>
<tr><td>0</td><td>0</td><td>No address hold wait cycles</td></tr>
<tr><td>0</td><td>1</td><td>1 clock cycle</td></tr>
<tr><td>1</td><td>0</td><td>2 clock cycles</td></tr>
<tr><td>1</td><td>1</td><td>3 clock cycles</td></tr>
</table></td></tr>
</table>

4.2.8 ICC0, ICC1 - Idle cycle configuration registers 0 and 1

The ICC registers are used to specify the number of idle cycles on the external bus. The number of idle cycles can be specified for each chip select area and in read cycles and write cycles.

Access This register can be read or written in 16-bit units.

Address $FFFF7214_H$: ICC0

$FFFF7216_H$: ICC1

Initial value 0000_H. This register is initialized by any reset.

Cautions

1. The number of idle cycles specified by the ICCm register (m = 0 or 1) is invalid during burst read and bus sizing cycles.
2. For the ICC0 register, only bits 15 to 8 can be set. Be sure to clear bits 7 to 0 to “0”.
3. For the ICC1 register, only bits 15 to 12 and 3 to 0 can be set. Be sure to clear bits 11 to 4 to “0”.

	15	14	13	12	11	10	9	8
ICC0	WIC31	WIC30	RIC31	RIC30	WIC21	WIC20	RIC21	RIC20
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	WIC11	WIC10	RIC11	RIC10	WIC01	WIC00	RIC01	RIC00
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

	15	14	13	12	11	10	9	8
ICC1	WIC71	WIC70	RIC71	RIC70	WIC61	WIC60	RIC61	RIC60
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	WIC51	WIC50	RIC51	RIC50	WIC41	WIC40	RIC41	RIC40
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-10 ICC0 and ICC1 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>13:12,
9:8,
5:4,
1:0</td><td>RICn1,
RICn0</td><td>Idle cycle configuration bits after a read cycle
Specify the number of idle cycles after a read cycle for each chip select area.
<table>
<tr><th>RICn1</th><th>RICn0</th><th>Number of idle cycles</th></tr>
<tr><td>0</td><td>0</td><td>No idle cycles</td></tr>
<tr><td>0</td><td>1</td><td>1 clock cycle</td></tr>
<tr><td>1</td><td>0</td><td>2 clock cycles</td></tr>
<tr><td>1</td><td>1</td><td>3 clock cycles</td></tr>
</table>
The setting specified for the RICn1 and RICn0 bits is enabled for read accesses in all bus modes and for all bus cycle types.</td></tr>
<tr><td>15:14,
11:10,
7:6,
3:2</td><td>WICn1,
WICn0</td><td>Idle cycle configuration bits after a write cycle
Specify the number of idle cycles after a write cycle for each chip select area.
<table>
<tr><th>WICn1</th><th>WICn0</th><th>Number of idle cycles</th></tr>
<tr><td>0</td><td>0</td><td>No idle cycles</td></tr>
<tr><td>0</td><td>1</td><td>1 clock cycle</td></tr>
<tr><td>1</td><td>0</td><td>2 clock cycles</td></tr>
<tr><td>1</td><td>1</td><td>3 clock cycles</td></tr>
</table>
The setting specified for the WICn1 and WICn0 bits is enabled for write accesses in all bus modes and for all bus cycle types.</td></tr>
</table>

4.2.9 EWC - External wait error configuration register

The EWC register is used to enable or disable the external wait error function for each chip select area.

The values specified for the EWC register becomes valid in the following bus cycles:

- SRAM bus cycle type (in separate bus mode)
- Multiplexed bus mode

Access This register can be read or written in 16-bit units.

Address $FFFF721A_H$

Initial value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	EW7	0	EW6	0	EW5	0	EW4
R	R/W	R	R/W	R	R/W	R	R/W
7	6	5	4	3	2	1	0
0	EW3	0	EW2	0	EW1	0	EW0
R	R/W	R	R/W	R	R/W	R	R/W

Table 4-11 EWC register contents

Bit position	Bit name	Function
14, 12, 10, 8, 6, 4, 2, 0	EWn	External wait error configuration bits Specify whether to enable or disable an external wait error in each chip select area. 0: Disable the external wait error. 1: Enable the external wait error. If this feature is enabled, the microcontroller forcibly cancels the wait state when it detects an external wait signal for 128 consecutive clock cycles, and the CPU generates a SYSERROR exception.

4.2.10 SEN - SDRAM enable control register

The SEN register is used to enable or disable SDRAM controller operations.

Access This register can be read or written in 8-bit units.

Adress $FFFF7220_H$

Initial value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	SE
R	R	R	R	R	R	R	R/W

Table 4-12 SEN register contents

Bit position	Bit name	Function
0	SE	Specify whether to enable or disable SDRAM controller operations. 0: SDRAM controller stopped (External bus cycles are not generated.) 1: Operation enabled

4.2.11 SDCR - SDRAM configuration register

The SDCR register is used to specify the number of wait cycles and the address multiplexed width when accessing SDRAM. When this register is written, a register write operation starts.

Access This register can be read or written in 16-bit units.

Address $FFFF7222_H$

Initial value $20C0_H$. This register is initialized by any reset.

Cautions

1. Before a register write operation starts, there are no SDRAM read or write cycles. Before accessing SDRAM, read the STR register values and make sure that the WCF bit is set to 1.
2. Before configuring the SDCR register, configure the SDRAM refresh control register (RFS).
3. After accessing SDRAM, to write to the SDCR register again, first clear the SEN.SE bit to 0.
4. When the SE bit is cleared to 0, be sure to write to the SDCR register before setting the SE bit to 1 again. At this time, if the SDCR register value does not need to be changed, write the same values.
5. Be sure to clear bits 11, 10, and 8 to “0”.
6. <R> The SDRAM configuration register can be written only once after reset is canceled. After writing, do not change the value. If the value is changed, the SDRAM might not be able to be accessed normally.

15	14	13	12	11	10	9	8
BST	LTM2	LTM1	LTM0	0	0	PDM	0
R/W	R/W	R/W	R/W	R	R	R/W	R
7	6	5	4	3	2	1	0
BCW1	BCW0	SSO1	SSO0	RAW1	RAW0	SAW1	SAW0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-13 SDCR register contents (1/2)

Bit position	Bit name	Function
15	BST	Enable or disable burst read. 0: Burst read disabled (initial value) 1: Burst read enabled
14 to 12	LTM2 to LTM0	Specify the CAS latency value applied when the SDRAM is read.

LTM2	LTM1	LTM0	CAS latency
0	0	0	1
0	0	1	2
0	1	0	3
Other than above			Setting prohibited

Table 4-13 SDCR register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>9</td><td>PDM</td><td>Specify this bit when the power down mode for SDRAM is used. If the PDM bit is set to 1, the CKE signal level is set to low when the SDRAM is not being accessed, causing the SDRAM to enter power down mode.
0: Do not use the power down mode. (Initial value)
1: Use the power down mode.</td></tr>
<tr><td>7, 6</td><td>BCW1, BCW0</td><td>Specify the number of wait cycles between the issuance of a bank active command and the issuance of a read/write command, or between the issuance of a precharge command and the issuance of a bank active command.
<table>
<tr><th>BCW1</th><th>BCW0</th><th>Number of wait cycles</th></tr>
<tr><td>0</td><td>0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>2</td></tr>
<tr><td>1</td><td>1</td><td>3 (initial value)</td></tr>
</table></td></tr>
<tr><td>5, 4</td><td>SSO1, SSO0</td><td>Specify the address shift width during on-page access judgment.
If the data bus width is set to 16 or 32 bits, the system does not use the lower addresses (A0, or, A1 and A0).
<table>
<tr><th>SSO1</th><th>SSO0</th><th>Address shift width</th></tr>
<tr><td>0</td><td>0</td><td>0 bits (data bus width: 8 bits) (initial value)</td></tr>
<tr><td>0</td><td>1</td><td>1 bit (data bus width: 16 bits)</td></tr>
<tr><td>1</td><td>0</td><td>2 bits (data bus width: 32 bits)</td></tr>
<tr><td>1</td><td>1</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>3, 2</td><td>RAW1, RAW0</td><td>Specify the row address width.
<table>
<tr><th>RAW1</th><th>RAW0</th><th>Row address width</th></tr>
<tr><td>0</td><td>0</td><td>11 bits (initial value)</td></tr>
<tr><td>0</td><td>1</td><td>12 bits</td></tr>
<tr><td>1</td><td>0</td><td>13 bits</td></tr>
<tr><td>1</td><td>1</td><td>14 bits</td></tr>
</table></td></tr>
<tr><td>1, 0</td><td>SAW1, SAW0</td><td>Specify the address multiplexed width (column address width) for accessing SDRAM.
<table>
<tr><th>SAW1</th><th>SAW0</th><th>Address multiplexed width (column address width)</th></tr>
<tr><td>0</td><td>0</td><td>8 bits (initial value)</td></tr>
<tr><td>0</td><td>1</td><td>9 bits</td></tr>
<tr><td>1</td><td>0</td><td>10 bits</td></tr>
<tr><td>1</td><td>1</td><td>11 bits</td></tr>
</table></td></tr>
</table>

Table 4-14 Row address output

Bit setting		Address pins																		
SAW 1	SAW 0	A28 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	a28 to a18	a25	a24	a23	a22	a21	a20	a19	a18	a17	a16	a15	a14	a13	a12	a11	a10	a9	a8
0	1	a28 to a18	a26	a25	a24	a23	a22	a21	a20	a19	a18	a17	a16	a15	a14	a13	a12	a11	a10	a9
1	0	a28 to a18	a27	a26	a25	a24	a23	a22	a21	a20	a19	a18	a17	a16	a15	a14	a13	a12	a11	a10
1	1	a28 to a18	a28	a27	a26	a25	a24	a23	a22	a21	a20	a19	a18	a17	a16	a15	a14	a13	a12	a11

Table 4-15 Column address output

(a) When any bank precharge command is issued

Bit setting		Address pins																		
SSO 1	SSO 0	A28 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	a28 to a18	a17	a16	a15	a14	a13	a12	a11	1	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0
0	1	a28 to a18	a17	a16	a15	a14	a13	a12	1	a10	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0
1	0	a28 to a18	a17	a16	a15	a14	a13	1	a11	a10	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0

(b) When a register write command is issued

Bit setting		Address pins																		
SSO 1	SSO 0	A28 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	LTM[2:0] + 1[a]			0	0	[b]	
0	1	0	0	0	0	0	0	0	0	1	0	0	LTM[2:0] + 1[a]			0	0	[b]		0
1	0	0	0	0	0	0	0	0	1	0	0	LTM[2:0] + 1[a]			0	0	[b]		0	0

a) LTM[2:0] bit value + 1

b) When burst read is enabled (SDCR.BST bit = 1): 11; otherwise: 00

(c) When a read or write command is issued

Bit setting		Address pins																		
SSO 1	SSO 0	A28 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	a28 to a18	a17	a16	a15	a14	a12	a11	a10	1[a]	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0
0	1	a28 to a18	a17	a16	a15	a14	a12	a11	1[a]	a10	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0
1	0	a28 to a18	a17	a16	a15	a14	a12	1[a]	a11	a10	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0

a) The value is 1 when read and 0 when written.

4.2.12 STR - SDRAM status register

The STR register indicates the SDRAM status.

Access This register is read-only, in 8-bit units.

Address $FFFF7224_H$

Initial value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	WBF	WCF
R	R	R	R	R	R	R	R

Table 4-16 STR register contents

Bit position	Bit name	Function
1	WBF	This bit indicates whether data is stored in the SDRAM controller write buffer. 0: No data is stored in write buffer (initial value) 1: Data is stored in the write buffer.
0	WCF	After the SDCR register is set, this bit indicates that a register write command for the SDRAM is complete. If the SEN.SE bit is cleared to 0, the WCF bit is also cleared to 0. 0: Setting not complete (initial value) 1: Setting complete

4.2.13 RFS - SDRAM refresh control register

The RFS register is used to specify the CBR refresh cycle and self refresh cycle for SDRAM. This register can also be used to specify the refresh cycle when SDRAM is initially configured (when a register write operation is executed).

Access This register can be read or written in 16-bit units.

Address $FFFF7226_H$

Initial value 0000_H. This register is initialized by any reset.

Caution To change the RFS register setting during SDRAM controller operation (SE bit = 1), use the following procedure:

1. Clear the SE bit of the SEN register to 0.
2. Clear the REN bit to 0.
3. While specifying new values for the RIN12 to RIN0 bits, set the REN bit to 1.
4. Set the SDCR register.
5. Make sure that the STR.WCF bit is 1, and then set the SE bit of the SEN register to 1.

When changing the refresh interval, specify a value that enables the refresh to be in time even during the change.

15	14	13	12	11	10	9	8
REN	0	0	RIN12	RIN11	RIN10	RIN9	RIN8
R/W	R	R	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RIN7	RIN6	RIN5	RIN4	RIN3	RIN2	RIN1	RIN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-17 RFS register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15</td><td>REN</td><td>Specify whether to enable or disable refreshing.
0: Refreshing disabled (initial value)
1: Refreshing enabled</td></tr>
<tr><td>12 to 0</td><td>RIN12 to RIN0</td><td>Specify the refresh interval factor.
<table>
<tr><th>RIN12</th><th>RIN11</th><th>...</th><th>RIN2</th><th>RIN1</th><th>RIN0</th><th>Interval factor (Ifac)</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (initial value)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>2</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>3</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>4</td></tr>
<tr><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>8,191</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>8,192</td></tr>
</table></td></tr>
</table>

Table 4-18 Example of SDRAM refresh intervals

Target refresh interval setting (μs)	Interval factor (Ifac)[a]
	When ϕ = 40 MHz
15	600 (15.0)
30	1,200 (30.0)
60	2,400 (60.0)

a) The values in parentheses are the calculated refresh intervals (μs).

Calculating the interval factor:

Ifac = X (μs)/(1/ϕ (MHz)); disregard any fractional component.

X: Target refresh interval setting

ϕ: SDRAM operating frequency

4.3 Bus Cycle Type Setting Funtion

In the separate bus mode, the bus cycle type listed below can be used. The BCT0 and BCT1 registers are used to specify a bus cycle type for each chip select area.

- SRAM bus cycle type

Note that the CS4 area is fixed to the SDRAM bus cycle type and cannot be changed.

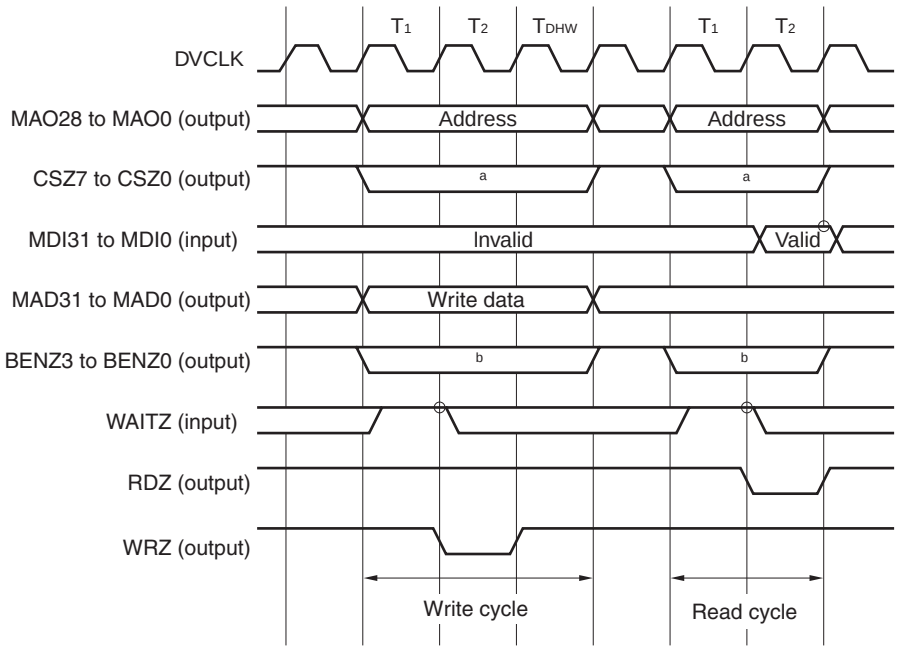
Moreover, in the multiplexed bus mode, it is possible to generate a bus cycle using the same signal line for address output and data input/output.

4.3.1 SRAM bus cycle type

The SRAM bus cycle is the most basic bus cycle in the separate bus mode. The microcontroller accesses the external memory using address and chip select signals, and the read strobe signal during the read cycle or the write strobe signal during the write cycle.

In the separate bus mode, this bus cycle is selected for all the chip select areas during initialization caused by a reset.

The read/write strobe cycle can be extended by up to 15 clock cycles by using the DWC register. External wait cycles are also enabled by these strobe signals.



a) Indicates the chip select area to be accessed.
b) Indicates the data size, which depends on the bus size specified for the BSC register.
For details, see *4.8 "Data Flow"*.

Figure 4-1 SRAM cycles (in separate bus mode, read/write)

The valid wait features in the SRAM cycle are listed below.

- Programmable data wait of up to 15 clock cycles specified by DWC register
- Data wait input by an external pin
- Insertion of idle cycles by using the ICC register
- Data hold wait of up to three clock cycles specified by using DHC register

4.3.2 SDRAM bus cycle type

This bus cycle is used when connecting SDR-SDRAM that conforms to JEDEC (it is not supported for Mobile SDRAM or SGRAM).

Only the CS4 area is allocated as an SDRAM area.The maximum supported SDRAM memory size is 512 Mb. The row address width corresponds to bits 11, 12, 13, and 14, and the column address width corresponds to bits 8, 9, 10, and 11. The maximum number of banks is four. Both little-endian and big-endian are supported.

When writing to SDRAM, single write access (burst length: 1) is always used. By buffering data into the SDRAM controller internal buffer in response to a write cycle request from the CPU (or DMAC), the CPU (or DMAC) can immediately perform the following processing. The buffer consists of eight stages, and, regardless of the SDRAM data bus width, can accumulate the data from up to eight writes. If even one item of data is accumulated in the write buffer, the STR.WBF flag is set to 1, and, when the write buffer empties, this flag is cleared to 0 when a precharge is executed. Write access to SDRAM continues until the write buffer empties.

When reading from SDRAM, the burst read function can be used. This function is enabled when the SDCR.BST bit is set to 1. When this function is enabled, eight sequential read accesses are made in response to a read cycle request from the CPU (or DMAC). If the data bus width is 8 bits, 64 bits including the requested address are sequentially accessed; if the data bus width is 16 bits, 128 bits including the requested address are sequentially accessed; and, if the data bus width is 32 bits, 256 bits including the requested address are sequentially accessed. The read data is saved in the SDRAM controller internal buffer, and then, when there is a read access request from the CPU (or DMAC), the address is compared to the address of the buffered data and, if the addresses are the same, the data is returned from the buffer without starting an external bus cycle. Any write access to SDRAM discards the read buffer data unconditionally. In addition, read requests from the CPU (or DMAC) are held pending until the eight data reads finish. (The generated eLB wait response output is retained.)

When reading SDRAM, the SDRAM controller issues commands in the order shown below according to whether the burst read function is enabled or disabled, regardless of which bank or page is accessed.

Burst read is enabled	Active command, auto precharge read command
Burst read is disabled	Active command, read command, precharge command

The CS4 area is a 64 MB space, but if an SDRAM smaller than 64 MB is connected, the remaining space is seen as a mirror of the actual SDRAM. For example, when a 32 MB SDRAM is connected as shown in *Figure 4-2*, if an attempt is made to access an address on the mirror side, on which the actual SDRAM does not exist, the corresponding SDRAM address is accessed. Note that even if burst read is enabled, reading an actual SDRAM address followed by reading the mirror side address always results in a read miss.

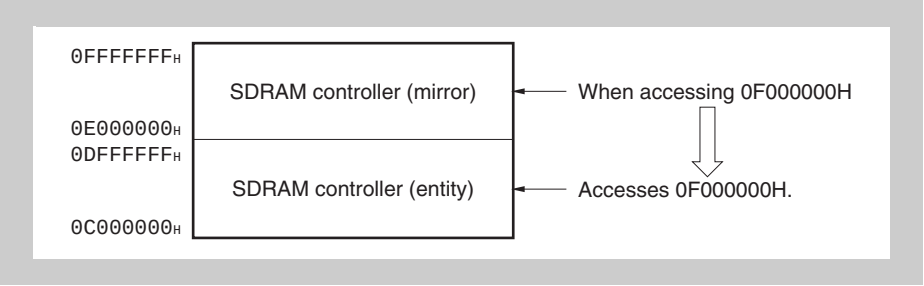


Figure 4-2 Example of accessing 32 MB SDRAM

The wait functions enabled for the SDRAM bus cycle type are as follows:

- Idle cycle insertion by the ICC register (but only the last cycle is enabled)
- RAS latency
- CAS latency

The address bus pins connected to the SDRAM are shown in the table below.

<R> **Table 4-19 Address bus pins connected to SDRAM (1/2)**

Row address width (RAW)	Column address width (SAW)	Address shift width (SSO)	A15	A14	A13	A12	A11	A10	...	A2	A1	A0	Bank[a]
11 bits (00)	8 bits (00)	8 bits (00)	–	–	–	–	–	a10	...	a2	a1	a0	A21 or later
11 bits (00)	8 bits (00)	16 bits (01)	–	–	–	–	a10	a9	...	a1	a0	–	A21 or later
11 bits (00)	8 bits (00)	32 bits (10)	–	–	–	a10	a9	a8	...	a0	–	–	A21 or later
12 bits (01)	8 bits (00)	8 bits (00)	–	–	–	–	a11	a10	...	a2	a1	a0	A21 or later
12 bits (01)	8 bits (00)	16 bits (01)	–	–	–	a11	a10	a9	...	a1	a0	–	A21 or later
12 bits (01)	8 bits (00)	32 bits (10)	–	–	a11	a10	a9	a8	...	a0	–	–	A22 or later
13 bits (10)	8 bits (00)	8 bits (00)	–	–	–	a12	a11	a10	...	a2	a1	a0	A21 or later
13 bits (10)	8 bits (00)	16 bits (01)	–	–	a12	a11	a10	a9	...	a1	a0	–	A22 or later
13 bits (10)	8 bits (00)	32 bits (10)	–	a12	a11	a10	a9	a8	...	a0	–	–	A23 or later
14 bits (11)	8 bits (00)	8 bits (00)	–	–	a13	a12	a11	a10	...	a2	a1	a0	A22 or later
14 bits (11)	8 bits (00)	16 bits (01)	–	a13	a12	a11	a10	a9	...	a1	a0	–	A23 or later
14 bits (11)	8 bits (00)	32 bits (10)	a13	a12	a11	a10	a9	a8	...	a0	–	–	A24 or later
11 bits (00)	9 bits (01)	8 bits (00)	–	–	–	–	–	a10	...	a2	a1	a0	A20 or later
11 bits (00)	9 bits (01)	16 bits (01)	–	–	–	–	a10	a9	...	a1	a0	–	A21 or later
11 bits (00)	9 bits (01)	32 bits (10)	–	–	–	a10	a9	a8	...	a0	–	–	A22 or later
12 bits (01)	9 bits (01)	8 bits (00)	–	–	–	–	a11	a10	...	a2	a1	a0	A21 or later
12 bits (01)	9 bits (01)	16 bits (01)	–	–	–	a11	a10	a9	...	a1	a0	–	A22 or later
12 bits (01)	9 bits (01)	32 bits (10)	–	–	a11	a10	a9	a8	...	a0	–	–	A23 or later
13 bits (10)	9 bits (01)	8 bits (00)	–	–	–	a12	a11	a10	...	a2	a1	a0	A22 or later
13 bits (10)	9 bits (01)	16 bits (01)	–	–	a12	a11	a10	a9	...	a1	a0	–	A23 or later
13 bits (10)	9 bits (01)	32 bits (10)	–	a12	a11	a10	a9	a8	...	a0	–	–	A24 or later
14 bits (11)	9 bits (01)	8 bits (00)	–	–	a13	a12	a11	a10	...	a2	a1	a0	A23 or later
14 bits (11)	9 bits (01)	16 bits (01)	–	a13	a12	a11	a10	a9	...	a1	a0	–	A24 or later
14 bits (11)	9 bits (01)	32 bits (10)	a13	a12	a11	a10	a9	a8	...	a0	–	–	A25 or later
11 bits (00)	10 bits (10)	8 bits (00)	–	–	–	–	–	a10	...	a2	a1	a0	A21 or later
11 bits (00)	10 bits (10)	16 bits (01)	–	–	–	–	a10	a9	...	a1	a0	–	A22 or later
11 bits (00)	10 bits (10)	32 bits (10)	–	–	–	a10	a9	a8	...	a0	–	–	A23 or later
12 bits (01)	10 bits (10)	8 bits (00)	–	–	–	–	a11	a10	...	a2	a1	a0	A22 or later
12 bits (01)	10 bits (10)	16 bits (01)	–	–	–	a11	a10	a9	...	a1	a0	–	A23 or later
12 bits (01)	10 bits (10)	32 bits (10)	–	–	a11	a10	a9	a8	...	a0	–	–	A24 or later
13 bits (10)	10 bits (10)	8 bits (00)	–	–	–	a12	a11	a10	...	a2	a1	a0	A23 or later
13 bits (10)	10 bits (10)	16 bits (01)	–	–	a12	a11	a10	a9	...	a1	a0	–	A24 or later
13 bits (10)	10 bits (10)	32 bits (10)	–	a12	a11	a10	a9	a8	...	a0	–	–	A25 or later
14 bits (11)	10 bits (10)	8 bits (00)	–	–	a13	a12	a11	a10	...	a2	a1	a0	A24 or later
14 bits (11)	10 bits (10)	16 bits (01)	–	a13	a12	a11	a10	a9	...	a1	a0	–	A25 or later
14 bits (11)	10 bits (10)	32 bits (10)	a13	a12	a11	a10	a9	a8	...	a0	–	–	A26 or later

<R> **Table 4-19 Address bus pins connected to SDRAM (2/2)**

Row address width (RAW)	Column address width (SAW)	Address shift width (SSO)	A15	A14	A13	A12	A11	A10	...	A2	A1	A0	Bank[a]
11 bits (00)	11 bits (11)	8 bits (00)	–	–	–	–	–	a10	...	a2	a1	a0	A22 or later
11 bits (00)	11 bits (11)	16 bits (01)	–	–	–	–	a10	a9	...	a1	a0	–	A23 or later
11 bits (00)	11 bits (11)	32 bits (10)	–	–	–	a10	a9	a8	...	a0	–	–	A24 or later
12 bits (01)	11 bits (11)	8 bits (00)	–	–	–	–	a11	a10	...	a2	a1	a0	A23 or later
12 bits (01)	11 bits (11)	16 bits (01)	–	–	–	a11	a10	a9	...	a1	a0	–	A24 or later
12 bits (01)	11 bits (11)	32 bits (10)	–	–	a11	a10	a9	a8	...	a0	–	–	A25 or later
13 bits (10)	11 bits (11)	8 bits (00)	–	–	–	a12	a11	a10	...	a2	a1	a0	A24 or later
13 bits (10)	11 bits (11)	16 bits (01)	–	–	a12	a11	a10	a9	...	a1	a0	–	A25 or later
13 bits (10)	11 bits (11)	32 bits (10)	–	a12	a11	a10	a9	a8	...	a0	–	–	A26 or later
14 bits (11)	11 bits (11)	8 bits (00)	–	–	a13	a12	a11	a10	...	a2	a1	a0	A25 or later
14 bits (11)	11 bits (11)	16 bits (01)	–	a13	a12	a11	a10	a9	...	a1	a0	–	A26 or later
14 bits (11)	11 bits (11)	32 bits (10)	a13	a12	a11	a10	a9	a8	...	a0	–	–	A27 or later

a) The bank address output signals output the address of the accessed space. For example, if the address output bus pins that can be used for outputting bank address are A21 or higher, A21 outputs “1” when A21 of the accessed address space is “1”.

Note RAW: SDCR.RAW[1:0] bit value
SAW: SDCR.SAW[1:0] bit value
SSO: SDCR.SSO[1:0] bit value
Axx: address output pins (MAOxx)
axx: address pins of connected SDRAM
–: not used

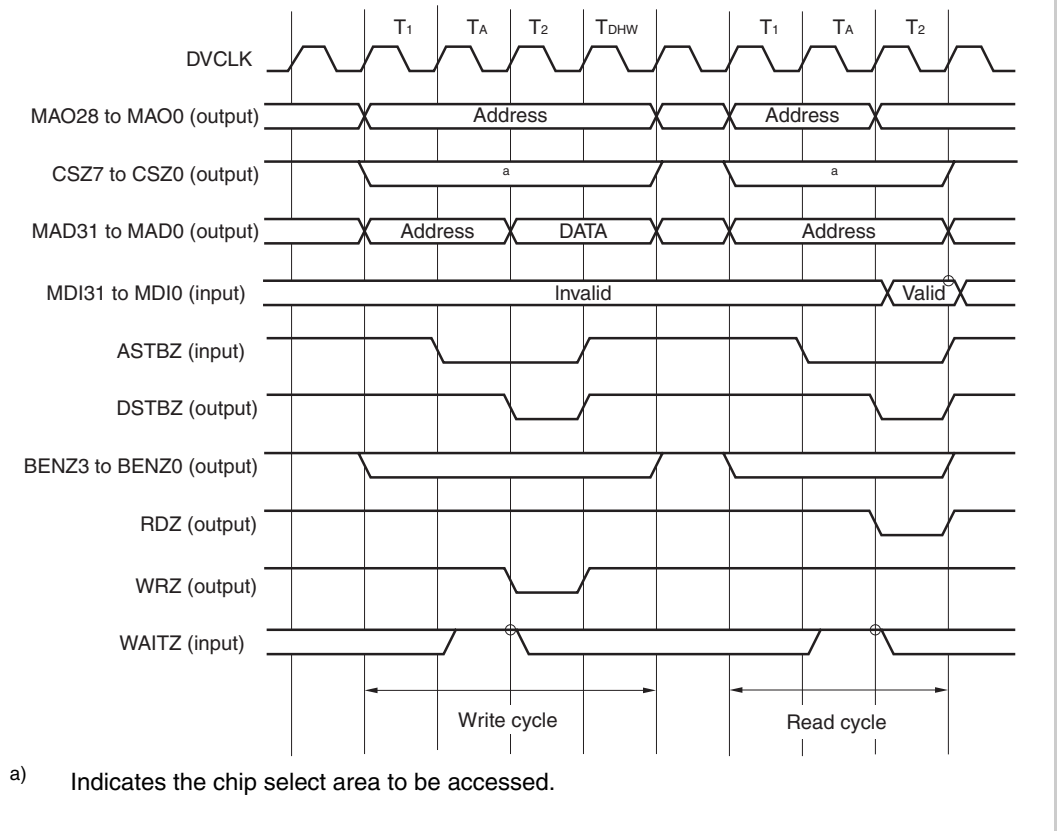
4.3.3 Multiplexed bus mode

The external bus access function can be used in the multiplexed bus mode.

In the multiplexed bus mode, external memory connection is achieved by switching address output and data input/output on the same signal line.

In addition to the address and data signals, chip select, address strobe, read strobe, and write strobe signals are used to control the external bus.

The multiplexed bus mode is used to connect an external ASIC in which the address bus and data bus are multiplexed, and to reduce the number of external signals.



a) Indicates the chip select area to be accessed.

Figure 4-3 Multiplexed bus mode (read cycle/write cycle)

The wait functions enabled for the multiplexed bus mode are as follows:

- Programmable data wait for up to 15 clock cycles specified by the DWC register
- Data wait input by an external pin
- Insertion of idle cycles for up to three clock cycles specified by the ICC register
- Data hold wait for up to three clock cycles specified by the DHC register
- Data setup wait for up to three clock cycles specified by the DHC register (only during write cycles)
- Address hold wait for up to three clock cycles specified by the AWC0 or AWC1 register

- Address setup wait for up to three clock cycles specified by the AWC0 or AWC1 register

Note When an external memory is connected using the multiplexed bus mode, an external data latch is required. For details, see *4.7 “Memory Connection Examples” on page 272*.

4.4 Bus Control Function

4.4.1 Chip select output function

The connected external memory area is managed divided into four chip select areas (CSn: n = 2 to 4, or 7), as shown in *Figure 4-4 "External memory map"*.

When a bus cycle is generated for the external bus, this microcontroller asserts the CSZn (n = 2 to 4, or 7) pin output corresponding to the accessed address (low level), along with outputting the accessed address from the MEMC0A[25:0] pins.

The various settings for the external bus, such as the bus size and number of wait/idle cycles, can all be specified for each chip select area.

By using these functions, different types of memory can be connected for each chip select area.

The allocation of the chip select areas is fixed by the system and cannot be reprogrammed.

The memory map is shown below.

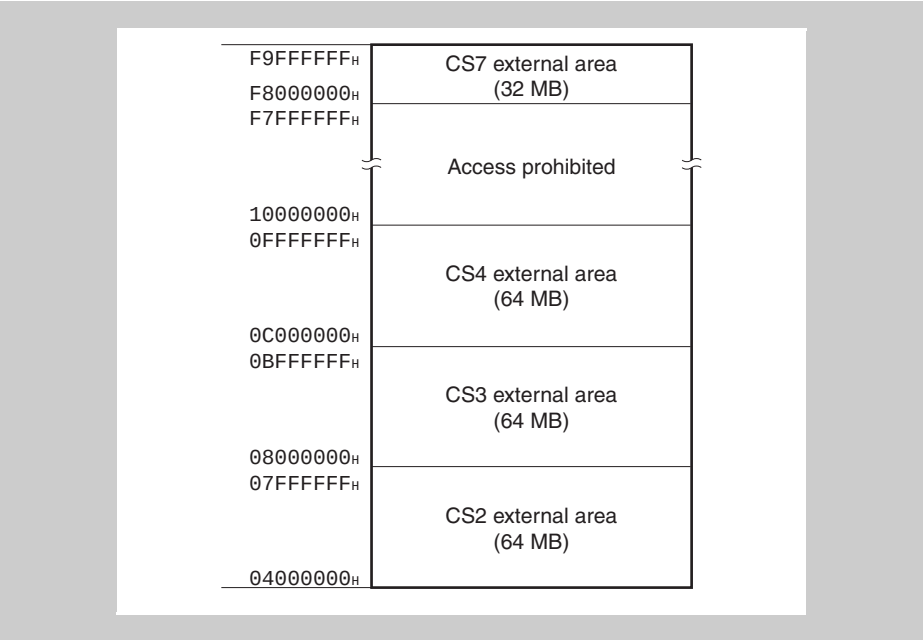


Figure 4-4 External memory map

4.4.2 Operation enable/disable setting function

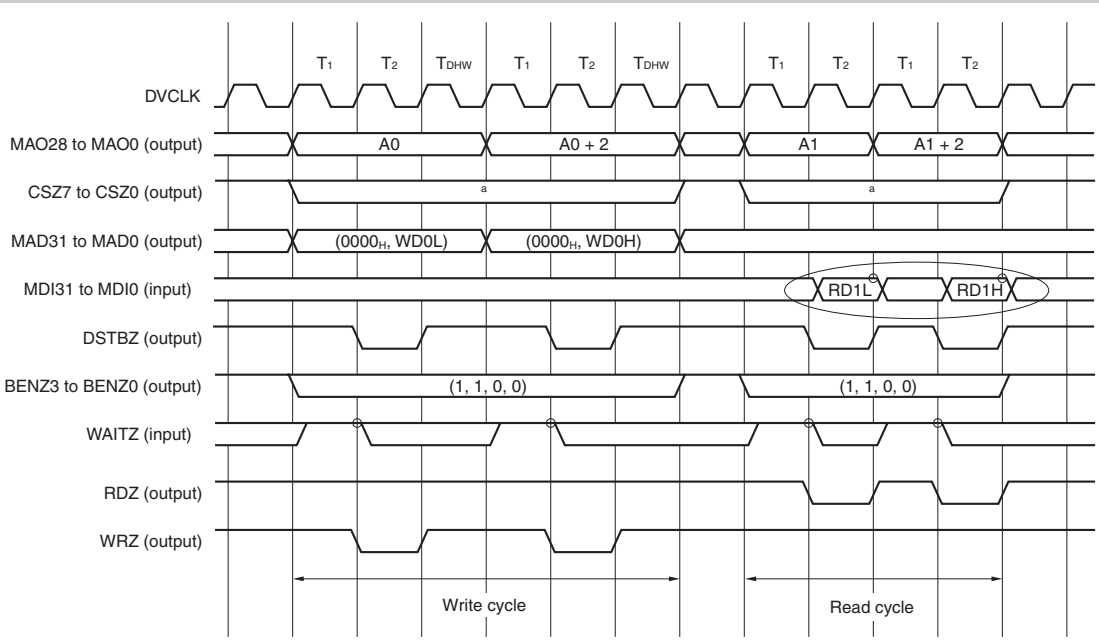
The operation can be enabled or disabled for each chip select area shown in *4.4.1 "Chip select output function" on page 245* by using the MEn bits of the BCT0 and BCT1 registers (n = 2 to 4, or 7).

If an access request is issued from the CPU (or DMAC) to a chip select area for which operation has been disabled by using this function, no external bus cycle is generated, the write value is ignored, and the read value becomes 00000000_H.

4.4.3 Bus size setting function

Access requests from the CPU (or DMAC) are executed by dividing the data in accordance with the bit width of the external bus at the access destination.

The bit width of the external bus can be selected from 32, 16, and 8 bits for each chip select area by using the BSC register.



a) Indicates the chip select area to be accessed.

Figure 4-5 SRAM cycle on a 16-bit bus

4.4.4 Data endian setting function

Either little-endian or big-endian can be selected as the data endianness of the external bus interface. This setting can be specified for each chip select area by using the DEC register. Initial setup by using input pins is not possible. In the initial status, little-endian is specified for all the chip select areas.

A chip select area whose endianness is specified as big-endian is accessed in big-endian mode.

This function can only be used for SRAM access.

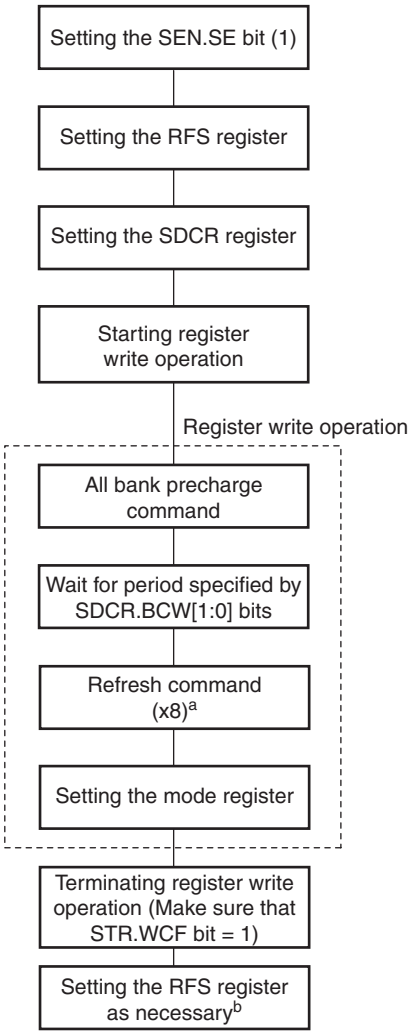
Cautions

1. This microcontroller does not support instruction fetching in big-endian mode.
2. Accessing a misaligned address in these areas is prohibited when big-endian is selected.

Note For details about the data flow for each external bus size and data size, see *4.8 "Data Flow" on page 279*.

4.4.5 SDRAM setting function

To use SDRAM, first set the SEN.SE bit, and then specify the refresh interval during initialization by using the SDRAM refresh control register (RFS). At this time, be sure to clear the REN bit to 0. Then specify the settings by using the SDRAM configuration registers (SDCR). When the SDCR register is written to, a register write operation is initiated. If eLB accesses the CS4 area before a register write operation is executed, the MEMC subsystem does not acknowledge the request nor return a response. When the register write operation finishes, the STR.WCF bit is set to 1. Before accessing SDRAM for the first time after initialization, be sure to read the STR register values and confirm that the WCF bit is set to 1.



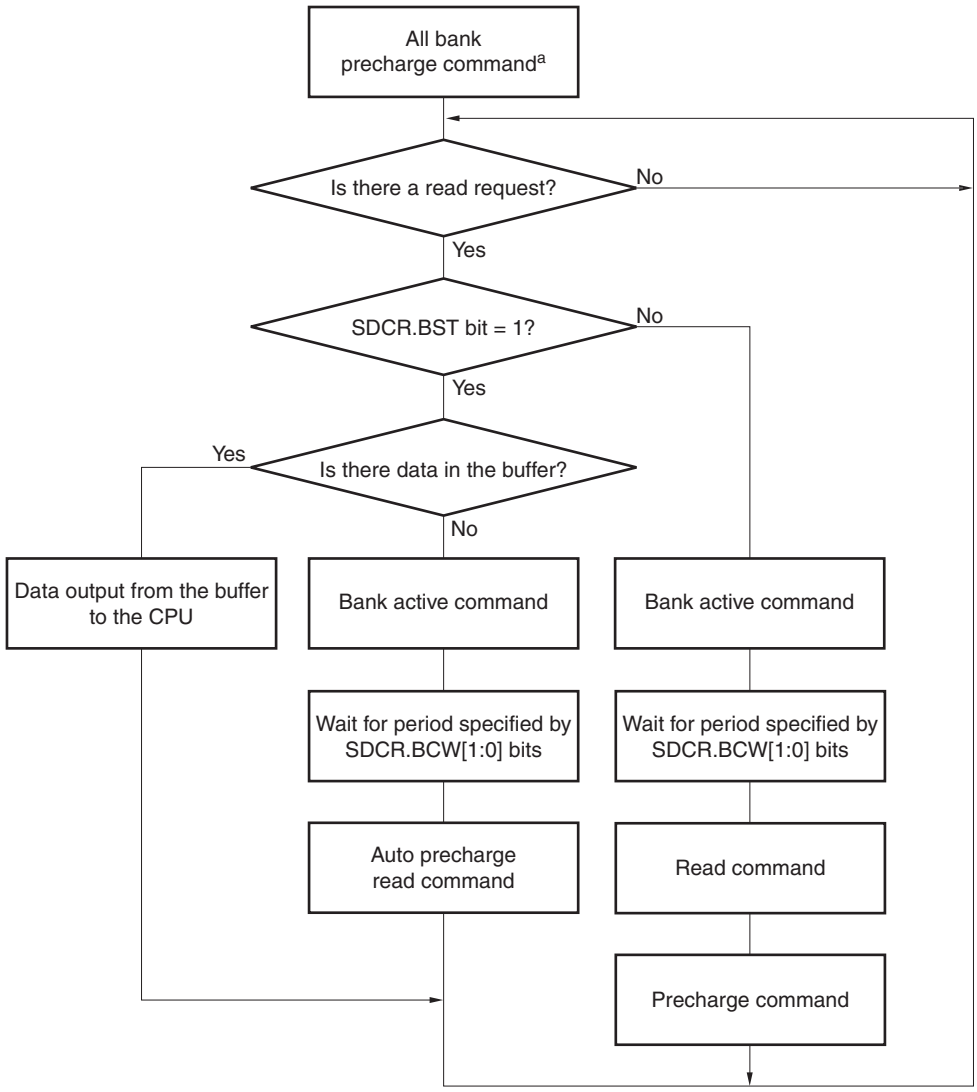
a) The RFS.RIN[12:0] bits are used to specify the interval between refresh commands during register write operations, and the interval between the eighth refresh command and mode register configuration.

b) Set the RFS.REN bit to 1 if a refresh cycle for SDRAM is required.At this time, if the refresh interval during a register write operation does not match the refresh interval during normal access, also change the RFS.RIN[12:0] bit value.

Figure 4-6 Flow of operations from initial register setup to register write

(1) SDRAM read access

When reading SDRAM, the burst read function can be used. If burst read is enabled, eight sequential read accesses are made, regardless of the data bus width. Read requests from the CPU (or DMAC) are held pending until the eight data reads finish. (The generated eLB wait response output is retained.) The read data is saved in the SDRAM controller internal buffer, and then, when there is a read access request from the CPU (or DMAC), the address is compared to the address of the buffered data and, if the addresses are the same, the data is returned from the buffer without starting an external bus cycle. Any write access to SDRAM discards the read buffer data unconditionally. If burst read is disabled, the addresses are not compared in response to a read access request from the CPU (or DMAC). Note that if burst read is enabled, the bank active command is followed by the auto precharge read command. If it is disabled, the bank active command is followed by the read command, and then the precharge command.



a) An all bank precharge command is always executed for SDRAM during a register write operation sequence after powering on or during a refresh sequence. An all bank precharge command is also executed after a bus hold period ends.

Figure 4-7 SDRAM read access status transitions

(2) Read buffer operation

The size of the SDRAM read buffer can be up to 256 bits. If the burst read function is enabled, this buffer can accumulate the data from up to eight reads. How to store data in the buffer depends on the data bus width. Operations for each data bus width are described below. If data is written to the write buffer even once, all the read buffer data is discarded. The read buffer data is also discarded if the bus enters the hold state.

(a) Data bus width (8 bits)

During 8-bit access, 64 bits, including the address of the data to be read, are stored in the buffer. If the lower 4 bits of the address of the target data are in the range from 0H to 7H, the read access starts at 0H, and eight burst read accesses are performed up to 7H, and the read data is stored in the buffer. If the lower 4 bits of the address of the target data are in the range from 8H to FH, the read access starts at 8H, and eight burst read accesses are performed up to FH, and the read data is stored in the buffer.

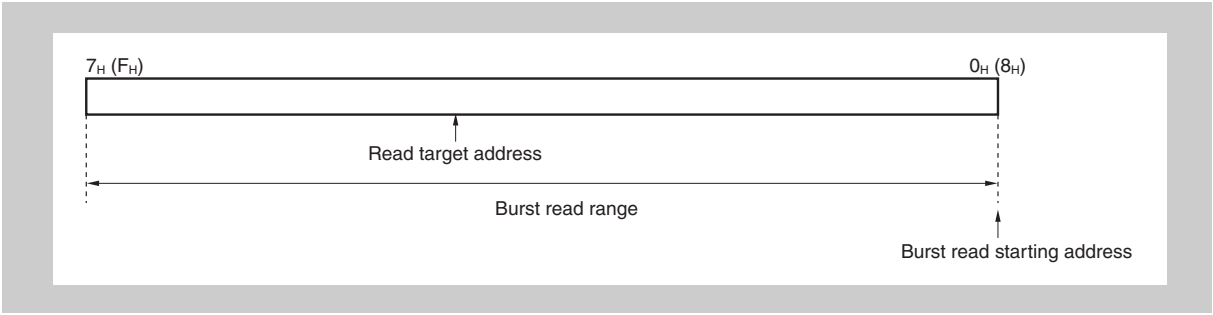


Figure 4-8 Data bus width (8 bits)

(b) Data bus width (16 bits)

During 16-bit access, 128 bits, including the address of the data to be read, are stored in the buffer. The lower 4 bits of the accessed address are monitored, and, within the 128-bit data range that includes the address of the target data, the read access starts at 0H, and eight burst read accesses are performed up to FH, and the read data is stored in the buffer.

Figure 4-9 Data bus width (16 bits)

(c) Data bus width (32 bits)

During 32-bit access, 256 bits, including the address of the data to be read, are stored in the buffer. The lower 5 bits of the accessed address are monitored, and, within the 256-bit data range that includes the address of the target data, the read access starts at 00H, and eight burst read accesses are performed up to 1FH, and the read data is stored in the buffer.

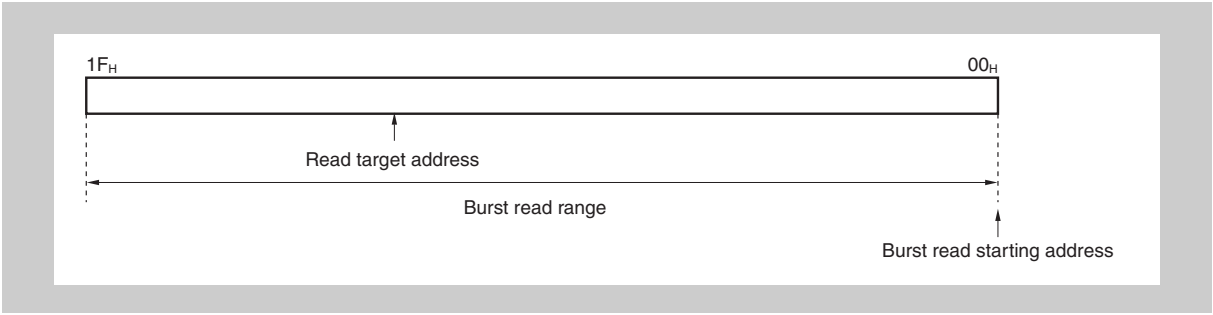
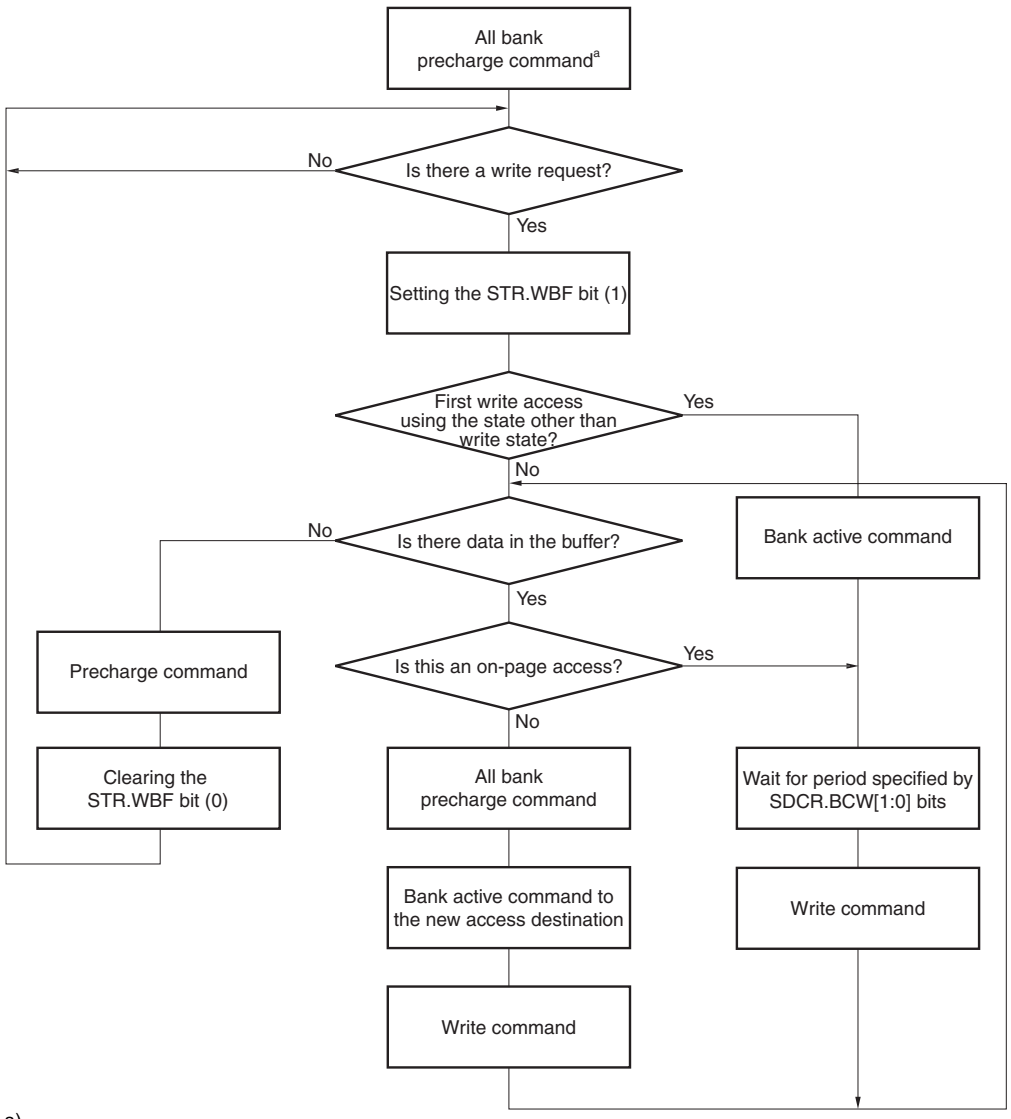


Figure 4-10 Data bus width (32 bits)

(3) SDRAM write access

When writing to SDRAM, single write access (burst length: 1) is always used. By buffering data into the SDRAM controller internal buffer in response to a write cycle request from the CPU (or DMAC), the CPU (or DMAC) can immediately perform the following processing. The buffer consists of eight stages, and, regardless of the SDRAM data bus width, can accumulate the data from up to eight writes. Write access to SDRAM continues until the write buffer empties.



a) An all bank precharge command is always issued for SDRAM during a register write operation sequence after powering on, or during a refresh sequence. An all bank precharge command is also issued after a bus hold period ends.

Figure 4-11 SDRAM write access status transitions

(4) Write buffer operation

The size of the SDRAM write buffer, which is an 8-stage FIFO buffer, is 256 bits. When the write buffer runs out of space, write access from the CPU (or DMAC) is held pending. Also, read access from the CPU (or DMAC) is held pending until the write buffer is empty. During a write cycle, a standby request, external bus hold request, or CBR refresh request may be made, but only a CBR refresh request can be acknowledged during the cycle.

Additionally, if data is stored in the write buffer, only a CBR refresh request has priority over the write command.

(5) CBR refresh function

The SDRAM controller starts a CBR refresh cycle at the refresh interval specified by using the RFS register. When a CBR refresh command is issued, if an external bus cycle has occurred, the CBR refresh command is held pending until this cycle ends. If an external bus hold request conflicts with a CBR refresh request, the external bus hold request is prioritized.

Note Because the refresh counter performs free-running operations, if a CBR refresh command is held pending, the interval until the next CBR refresh cycle is shortened.

(6) Self refresh function

When the CPU enters standby mode, the SDRAM controller performs the following procedure to start an SDRAM self refresh cycle:

1. If the MEMC is accessing an external bus, the MEMC sends a standby request signal to the SDRAM controller after the access finishes.
   Proceed to *4*.
   If the MEMC is not accessing an external bus, the MEMC immediately sends a standby request signal to the SDRAM controller.
2. If the SDRAM controller is reading data from SDRAM, the MEMC finishes the read access.
   – If the burst read function is enabled, the MEMC finishes eight read accesses.
   – If the timing at which a standby request conflicts with a read request to SDRAM, the reading SDRAM is prioritized.
3. If the SDRAM controller is writing data to SDRAM, the MEMC finishes the writing to SDRAM.
   – All data accumulated in the write buffer is written to SDRAM.
   – If asserting a standby request conflicts with a read request to SDRAM, the writing to SDRAM is prioritized.
4. The all bank precharge command, NOP command, and self refresh command are issued.
5. SDRAM enters the self refresh status.
6. Standby mode is entered.
7. Processing to release the self refresh begins. (The NOP command is issued, and (BCW setting value x 4) clock wait cycles are inserted.)
8. The SDRAM self refresh status is released.
9. The system returns to the normal status.

If an external bus hold request conflicts with a request to enter standby mode, the request to enter standby mode is prioritized.

4.5 Wait Functions

Wait functions are listed below.

Table 4-20 Wait functions

Wait functions			Data wait		Data hold wait	Data setup wait	Address wait	Idle	CAS latency	RAS latency
			Programmable	External wait						
Separate bus mode	SRAM bus cycle type	Read	√	√				√		
		Write	√	√	√			√		
	SDRAM bus cycle type	Read						√	√	√
		Write						√		√
Multiplexed bus mode		Read	√	√			√	√		
		Write	√	√	√	√	√	√		
Setting register			DWC0 DWC1	–	DHC	DSC	AWC	ICC0 ICC1	SDCR	SDCR
Maximum number of wait cycles			15	–	3	3	3	3	3	3

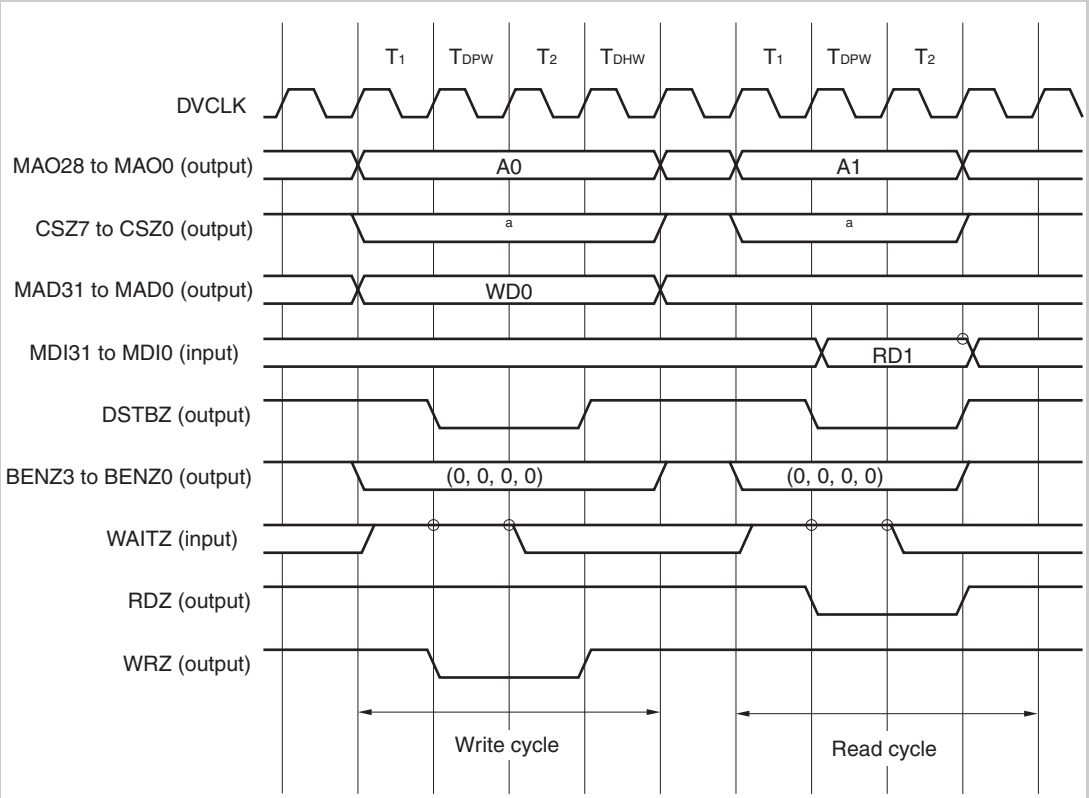
4.5.1 Programmable data wait function

This wait function is for delaying the data latch timing by extending the read strobe and write strobe periods.

This function is enabled during any write access and when the first data is transferred to SRAM.

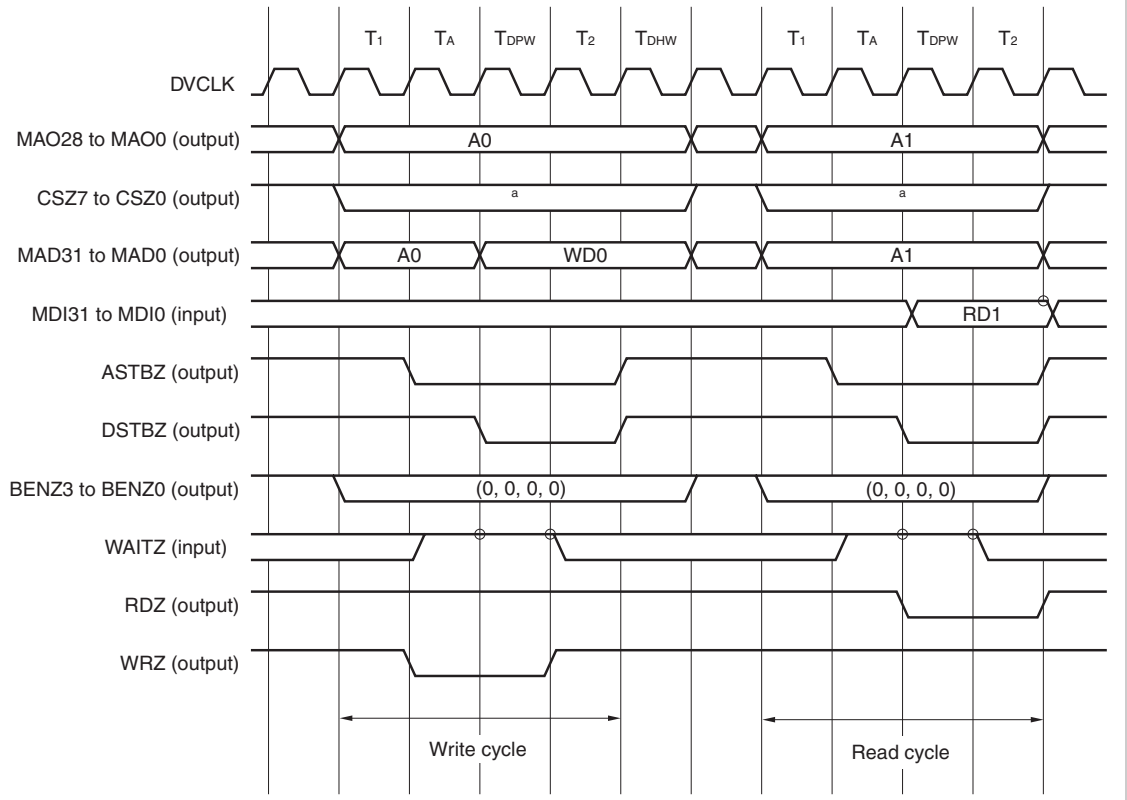
Up to 15 cycles can be inserted.

Setting individual chip select areas by using the DWC0 register is possible.



a) Indicates the chip select area to be accessed.

Figure 4-12 Programmable data wait in separate bus mode



a) Indicates the chip select area to be accessed.

Figure 4-13 Programmable data wait in multiplexed bus mode

4.5.2 External wait functions

If the separate bus mode or the multiplexed bus mode is selected as the bus cycle type, data wait cycles of any length can be inserted by using the WAITZ pin.

The WAITZ pin input level is sampled immediately after the T_A, T_1 cycles and the T_{DPW}, T_{DEW} cycles end.

Data wait cycles obtained by ORing the programmable data wait cycle specified by data wait control register 0 (DWC0 register) and the external wait specified by the WAITZ pin input, are inserted.

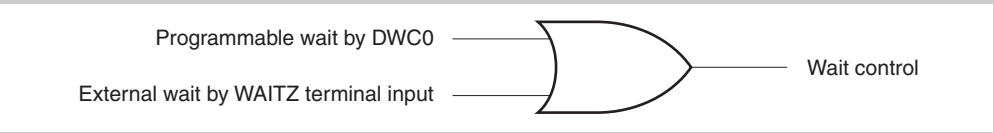
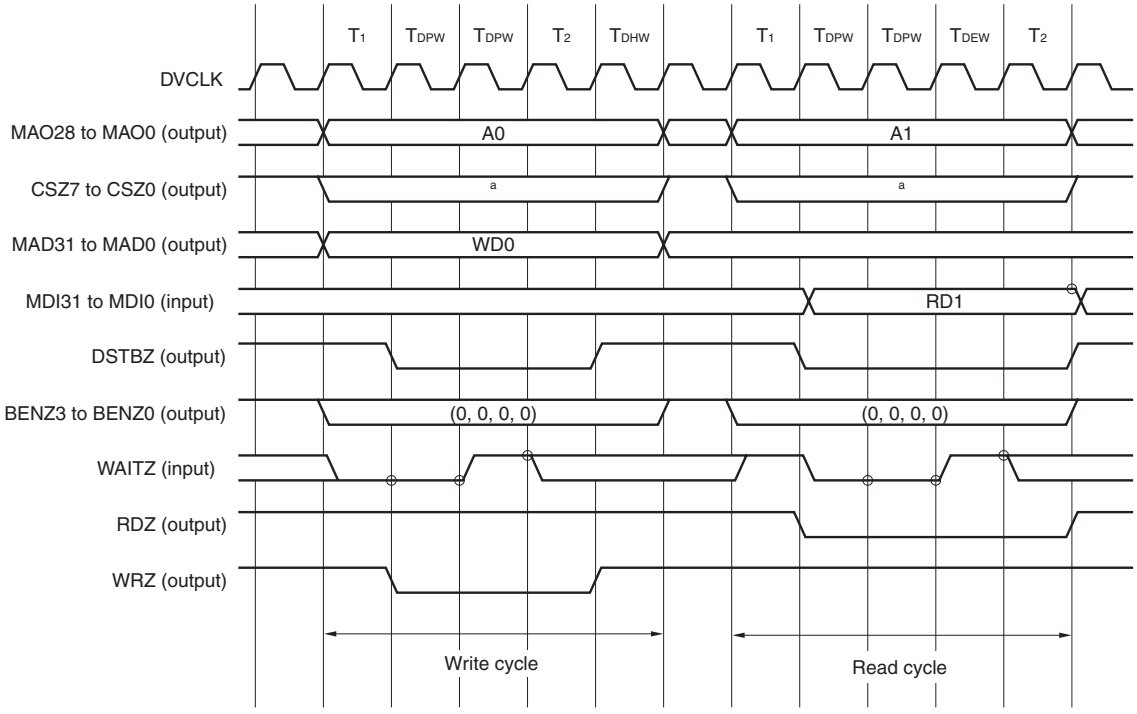


Figure 4-14 Internal data wait generator



a) Indicates the chip select area to be accessed.

Figure 4-15 Relationship between external data wait and programmable data wait (when DWC = 2)

4.5.3 External wait error detection

This microcontroller can forcibly cancel the data wait cycles if an external wait is continuously input for 128 clock cycles and report the error to the CPU (or DMAC) that issued the access request, by setting the EWn bit of the external wait error setting register (EWC). A SysError exception occurs in the CPU at this time.

Using this feature, even if an unforeseen problem occurs at the WAITZ input pin, the system does not hang up and exception handling can be executed for the detected anomaly.

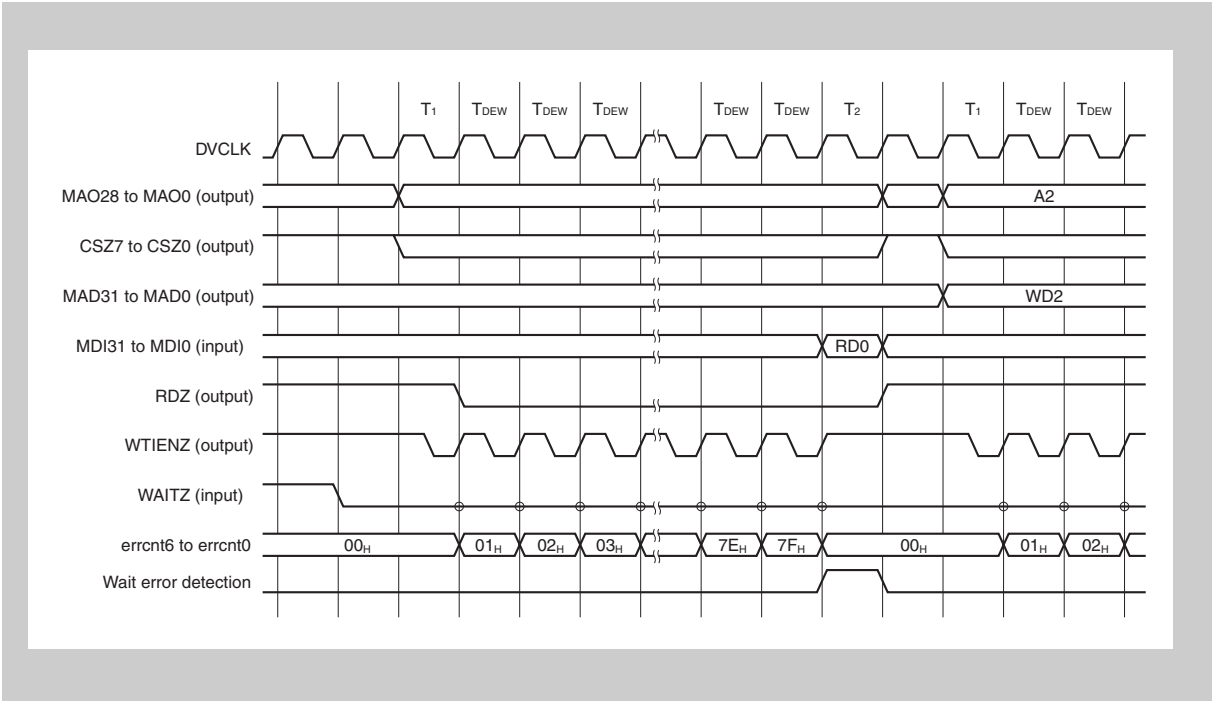


Figure 4-16 Operation timing when an external wait error is detected

Following the occurrence of an error at the CPU that issued the access request, the external wait error detector is initialized, and all transfer requests, including data wait requests from the WAITZ input pin, are processed normally.

At this time, when an external wait is continuously input for 128 clock cycles again, the data wait cycle is forcibly canceled, and an error is reported to the CPU that issued the access request.

4.5.4 Data setup wait function

This function is used to insert a wait cycle prior to the transfer state to secure the setup time for the data write strobe signal.

This function is enabled only during write cycles in the multiplexed bus mode.

Up to three cycles can be inserted.

This setting can be specified for each chip select area by using the DSC register. The initial status is no waits for any of the chip select areas.

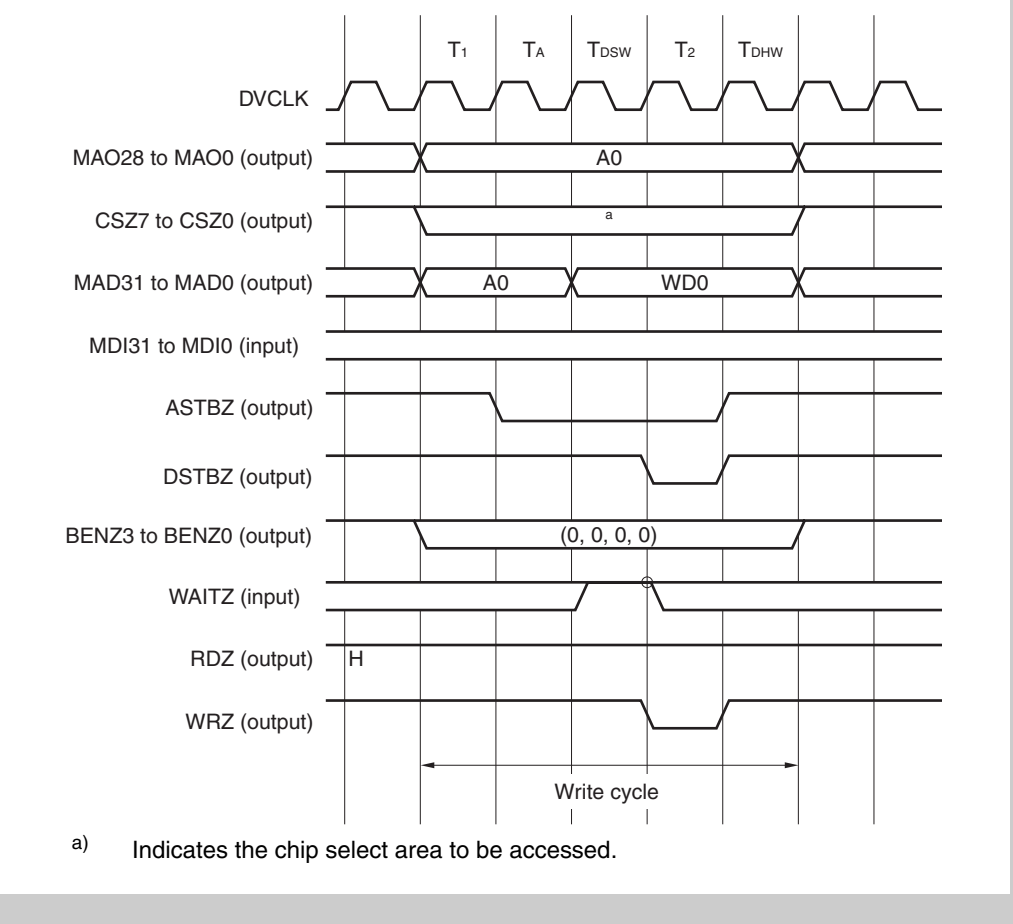


Figure 4-17 Data setup wait

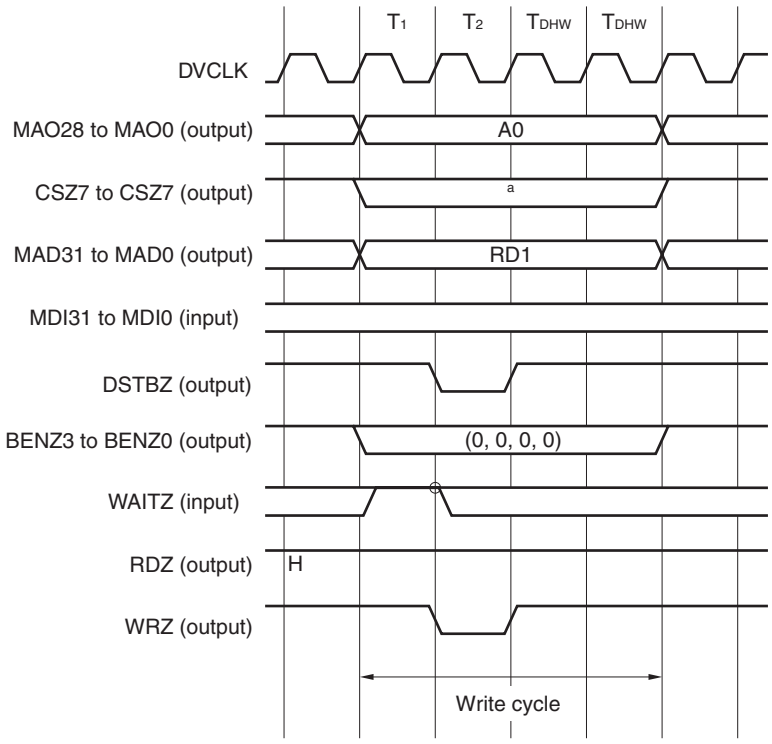
4.5.5 Data hold wait function

This function is used to insert a wait cycle for the state following the rising edge of the write strobe signal to secure the hold time for the data write strobe signal.

This function is enabled only during the write cycle for all bus cycle types.

This microcontroller always inserts one data hold wait state when a write cycle occurs. This data hold wait cycle is extended by up to three clock cycles, as specified for the DHC register, allowing insertion of four clock cycles.

The number of extended data hold wait cycles can be specified for each chip select area by using the DHC register. The initial status is no wait extension for any of the chip select areas (one data hold wait cycle).



a) Indicates the chip select area to be accessed.

Figure 4-18 Data hold wait in separate bus mode

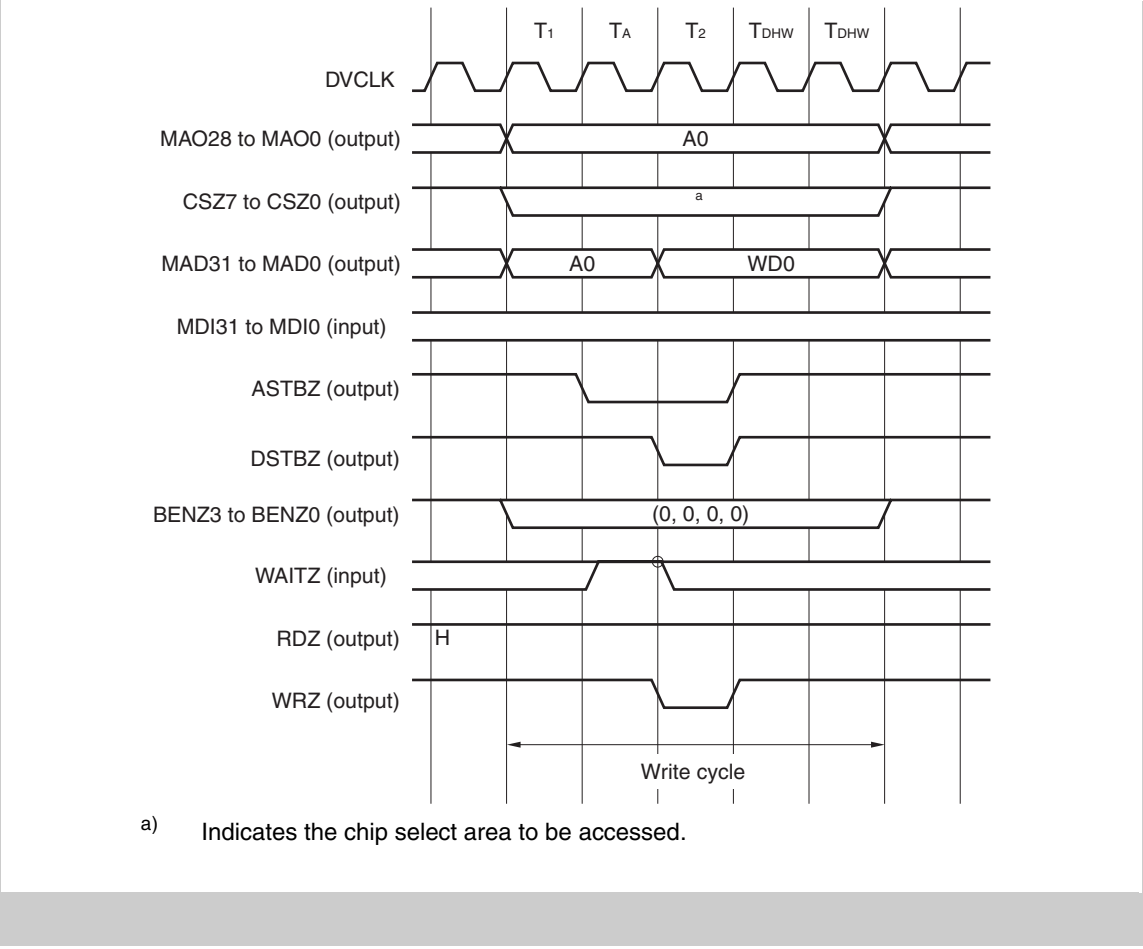


Figure 4-19 Data hold wait in multiplexed bus mode

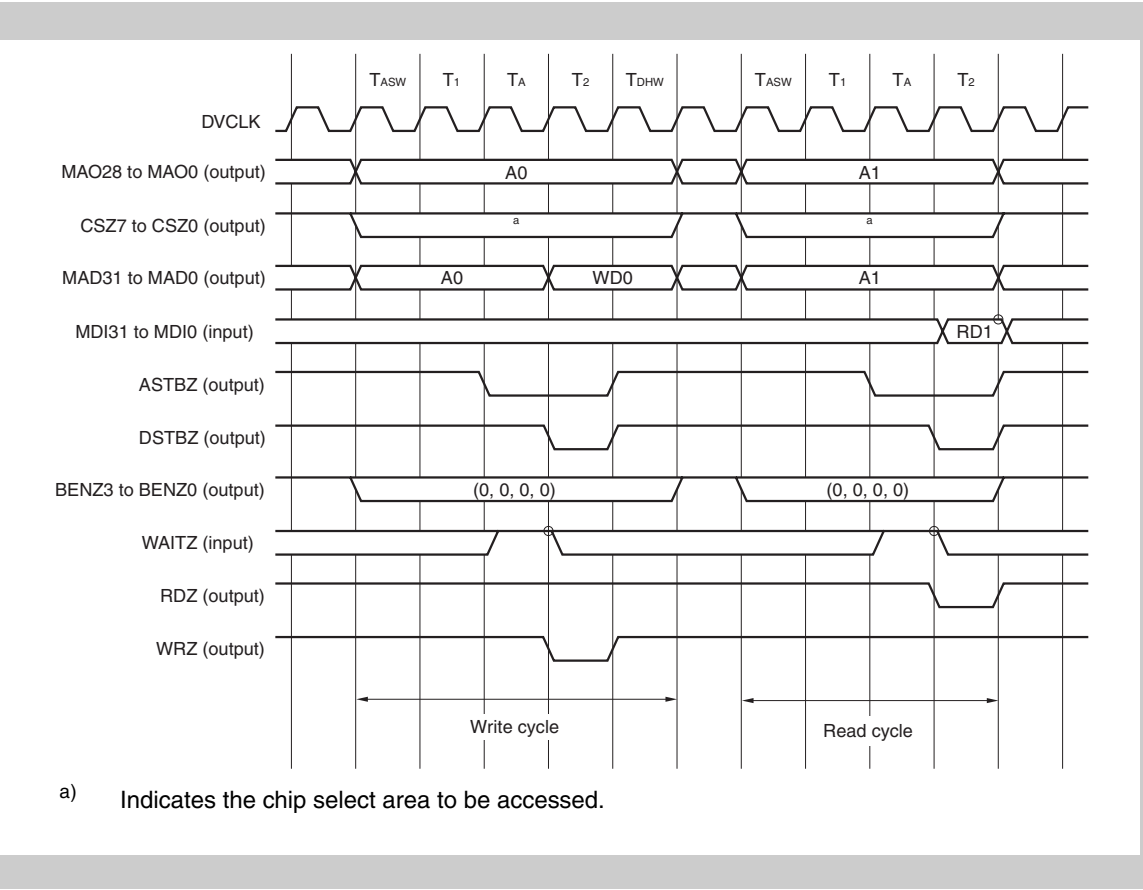
4.5.6 Address setup wait function

This function is used to insert a wait cycle prior to the address transfer state to secure the setup time for the address strobe signal in the multiplexed bus mode.

This function is enabled only in the multiplexed bus mode.

Up to three cycles can be inserted.

This setting can be specified for each chip select area by using the AWC0 and AWC1 registers.



a) Indicates the chip select area to be accessed.

Figure 4-20 Address setup wait

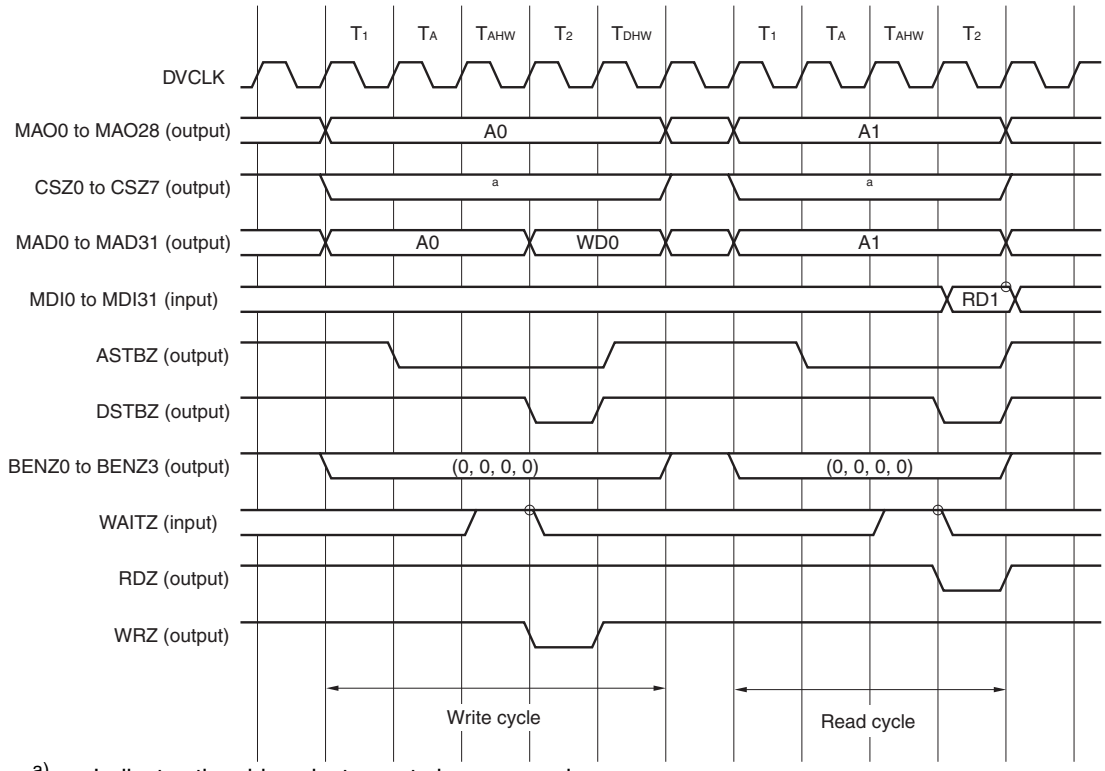
4.5.7 Address hold wait function

This function is used to insert a wait cycle following the address transfer state to secure the hold time for the address strobe signal in the multiplexed bus mode.

This function is enabled only in the multiplexed bus mode.

Up to three cycles can be inserted.

This setting can be specified for each chip select area by using the AWC0 and AWC1 registers.



a) Indicates the chip select area to be accessed.

Figure 4-21 Address hold wait

4.5.8 Idle insertion function

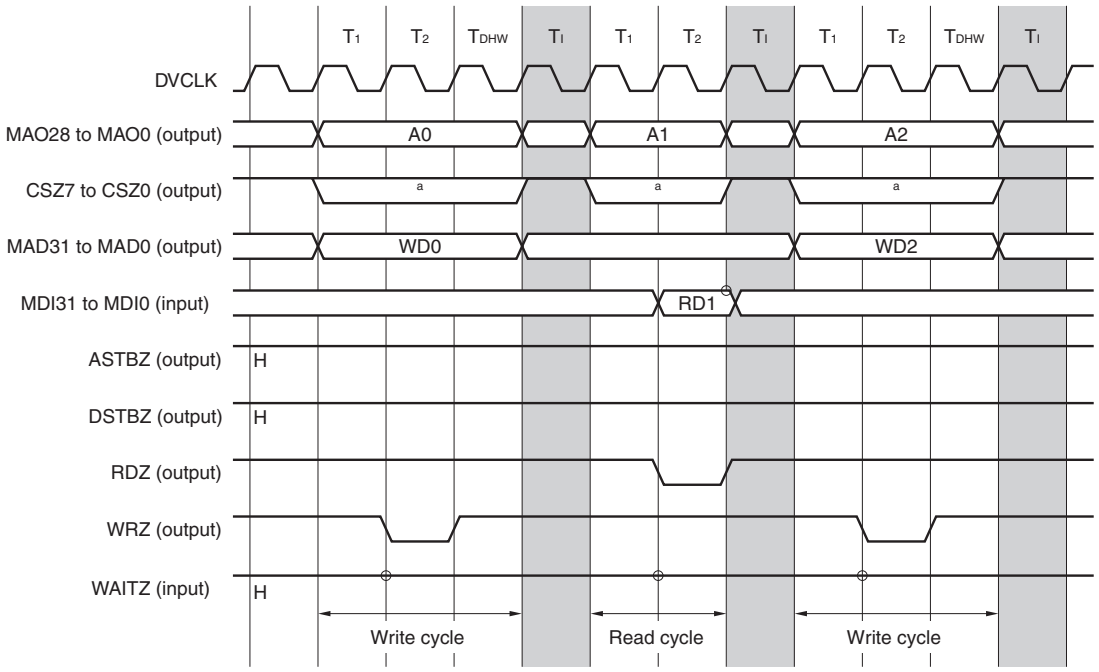
This function is used to insert an idle state following the last state of each cycle to prevent bus conflicts between cycles.

Up to three cycles can be inserted for all memory types.

This function can be specified independently after a read cycle or a write cycle for each chip select area by using the ICC0 and ICC1 registers.

The initial status is no idle cycles for any of the chip select areas.

Caution After a bus cycle ends, it takes one cycle until the subsequent bus cycles from the CPU (or DMAC) occur, regardless of the idle cycle setting. Therefore, a one-cycle interval occurs between bus cycles even if the setting is specified to be no idle cycles.



a) Indicates the chip select area to be accessed.

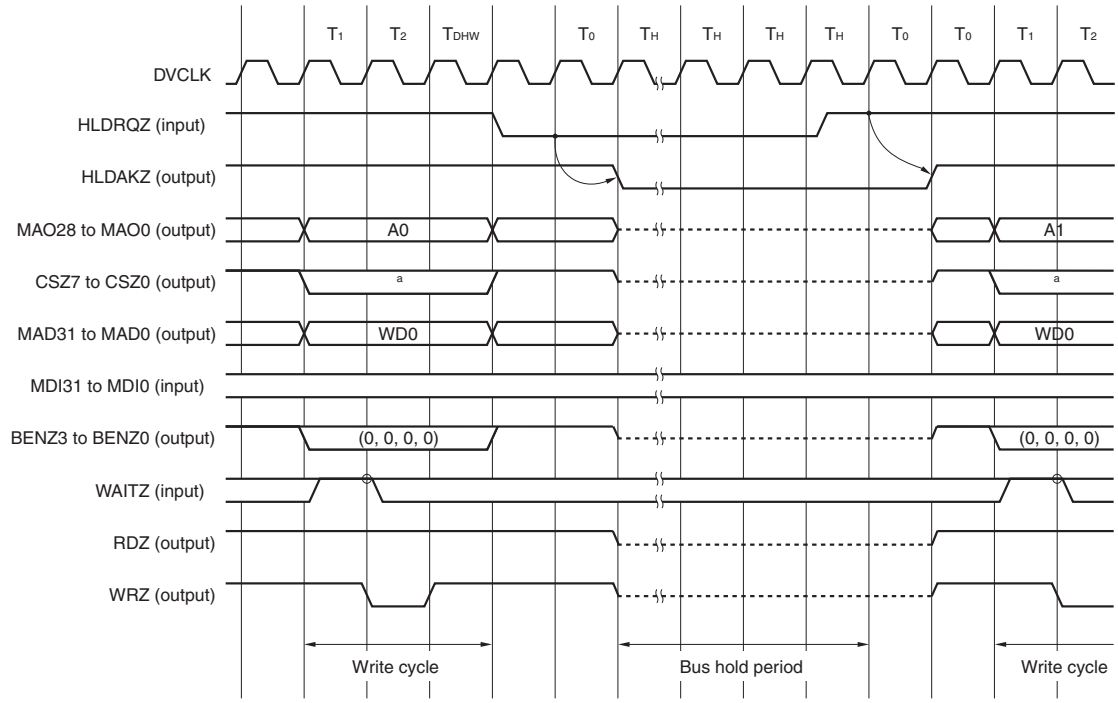
Figure 4-22 Idle cycles

4.6 Bus Hold Function

When the HLDRQZ pin input signal is asserted (low level) and a bus hold request is received from outside, this microcontroller enters the bus hold cycle after the current external bus cycle ends. When the bus hold cycle is entered, the active level is output from the HLDAKZ pin.

When the HLDRQZ pin is asserted (high level), the external bus goes into the normal operation mode.

Caution If the bus hold function is used, the MAD0 to MAD31 pins are forced to enter input mode after a bus hold request is received. To prevent through current, the option to connect the on-chip pull-up/pull-down resistors, or external pull-up/ pull-down resistors, must be used.



a) Indicates the chip select area to be accessed.

Note The broken lines indicate high impedance.

Figure 4-23 Bus hold function

During execution of bus cycles involving lock, such as read-modify-write from the CPU, the transition to the bus hold cycle is held pending until the bus cycle for that lock is completed.

The HLDRQZ pin input is sampled at the rising edge of the external bus operating clock.

When an external bus hold request is issued, the bank information internally held by the MEMC is discarded and, after the external bus hold cycle is canceled, the MEMC cycle always starts when a bank active command is issued.In addition, if the external bus master accesses SDRAM, the MEMC cycle must always start when an all bank precharge command is issued.

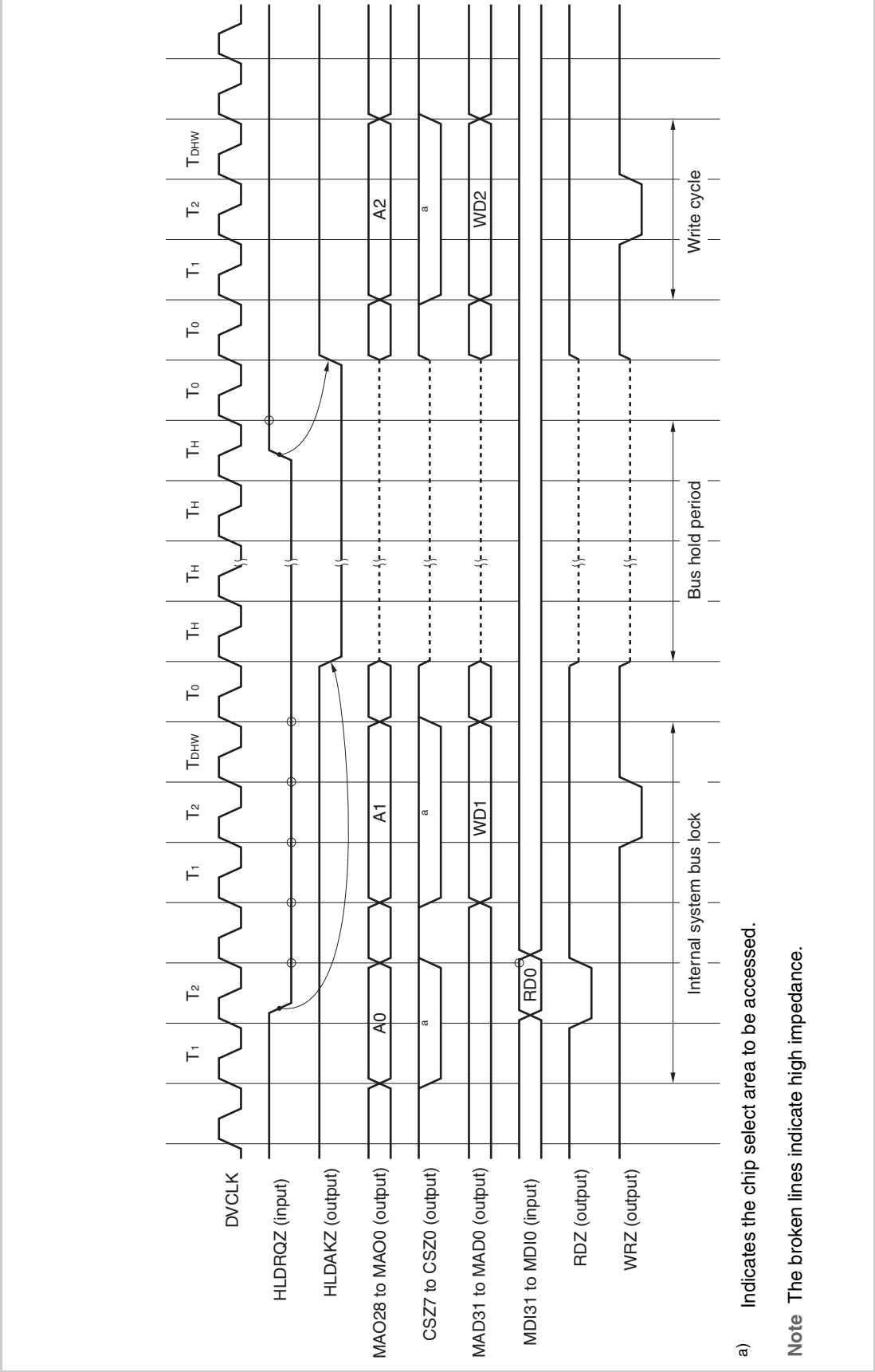


Figure 4-24 Delay of bus hold cycle due to internal system bus lockup

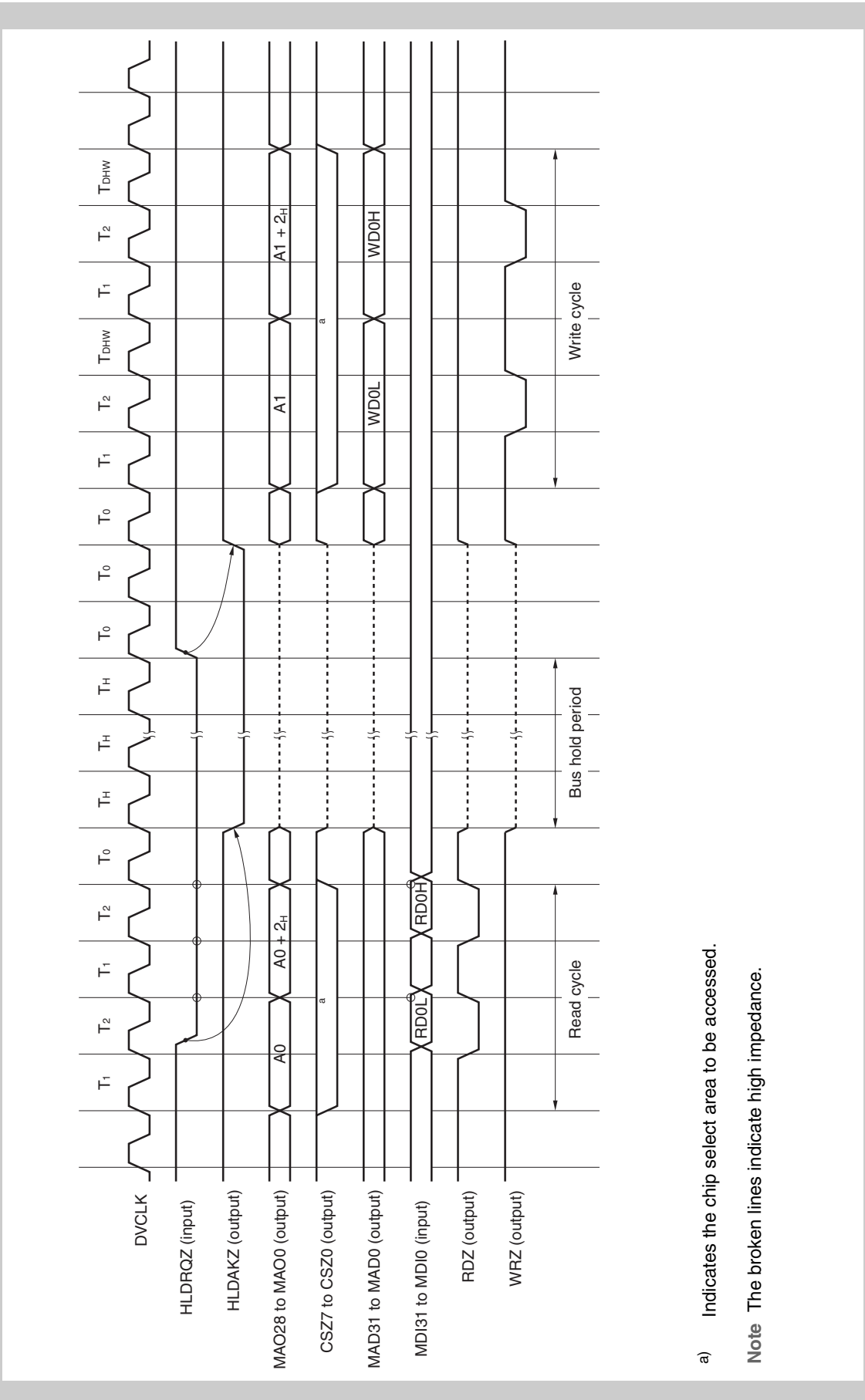
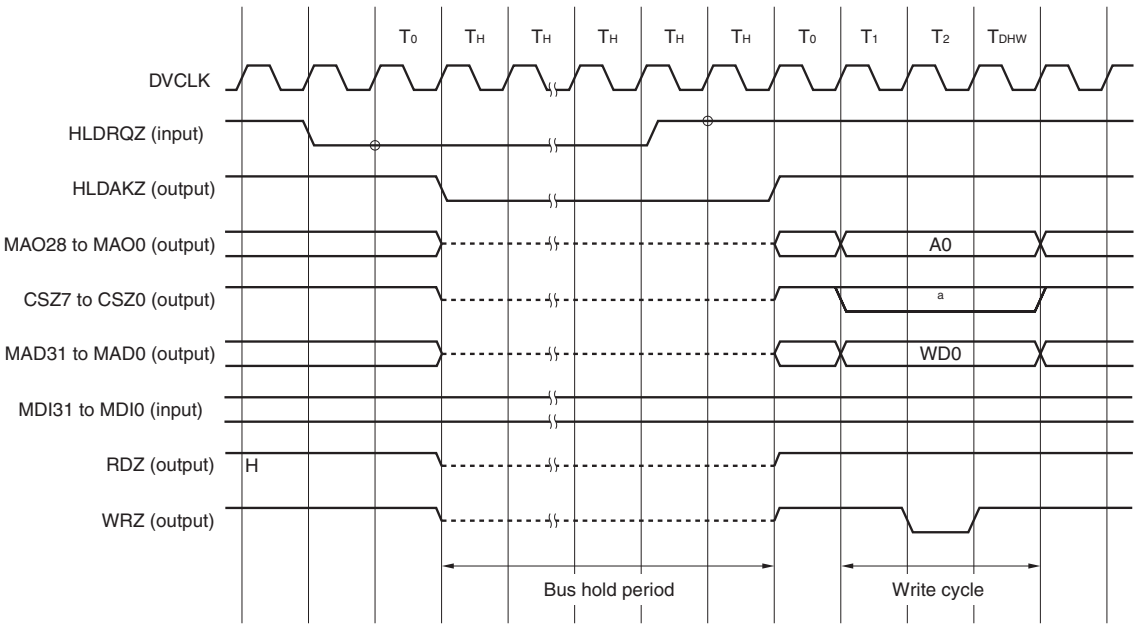


Figure 4-25 Delay of bus hold cycle due to bus sizing

If a bus hold request and a transfer request from the CPU occur at the same time, a wait response is returned for the transfer request and the transition to the bus hold cycle is executed first.



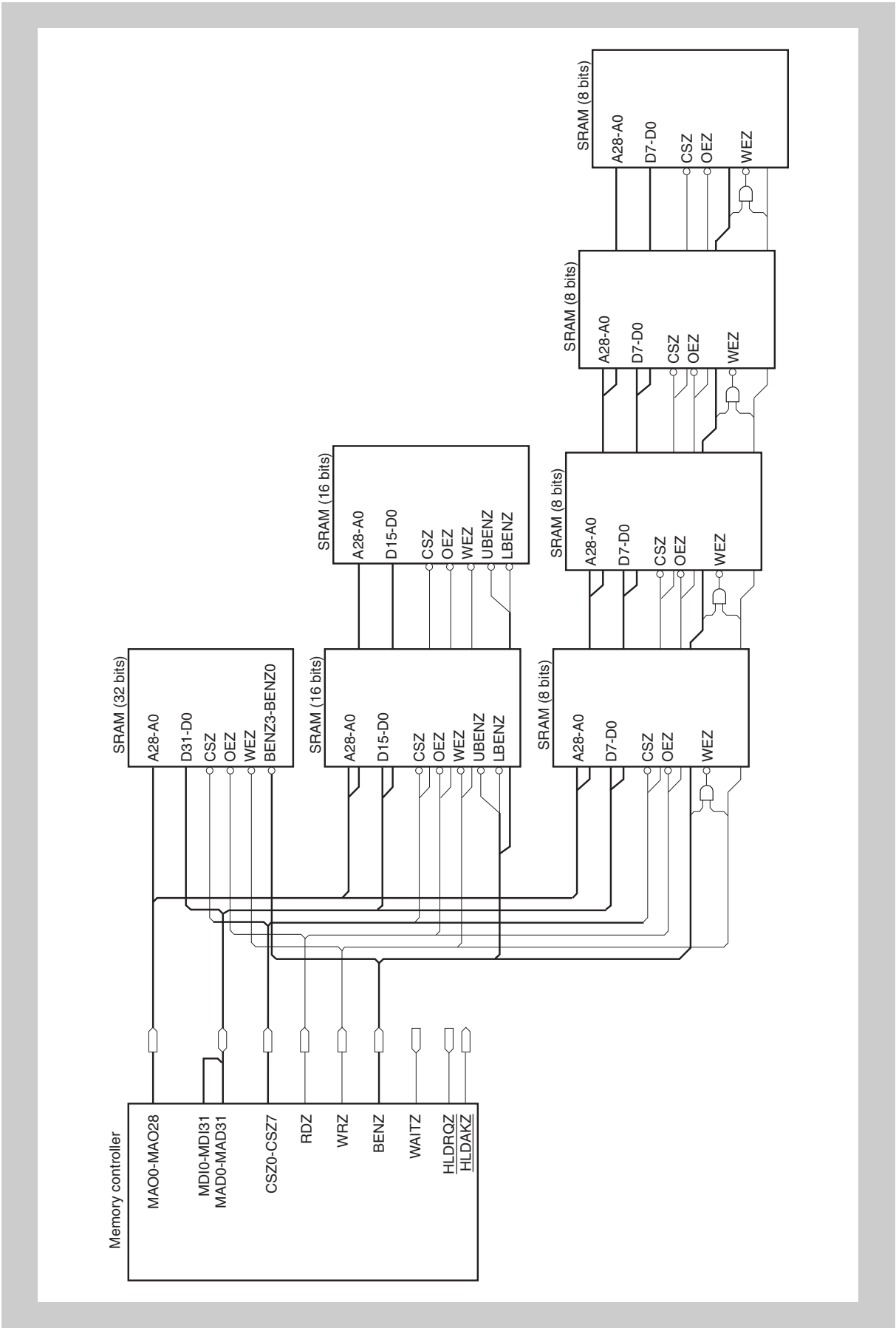
a) Indicates the chip select area to be accessed.

Note The broken lines indicate high impedance.

Figure 4-26 Operation when a conflict occurs between a bus hold request and a transfer request

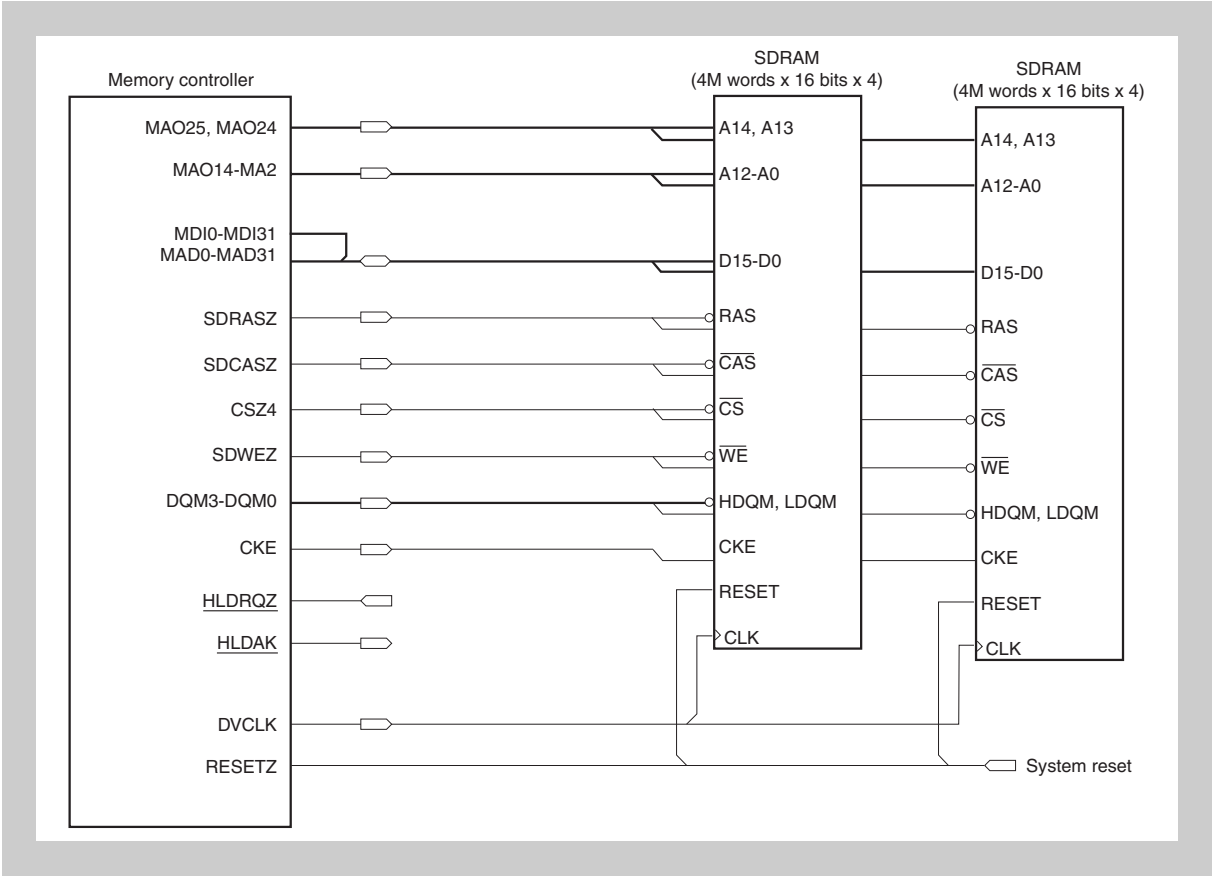
4.7 Memory Connection Examples

4.7.1 SRAM connection example



<R> **Figure 4-27 SRAM connection example**

4.7.2 SDRAM connection example



<R> **Figure 4-28 512 Mb SDRAM connection example (256-bit SDRAM (4M words × 16 bits × 4 banks) × 2)**

(1) Address output and SDRAM connection

This section describes the settings of the SDRAM configuration register (SDCR), the physical addresses, the address output from the MEMC, and the connection between the MEMC and SDRAM for each data bus width (8 bits, 16 bits, and 32 bits).

(a) When the data bus width is 8 bits

The following shows an example of how to connect 64 Mb SDRAM (2M words × 8 bits × 4 banks) when the data bus width is 8 bits.

- SDCR register settings

  SSO1 and SSO0 = 00: Data bus width = 8 bits
  RAW1 and RAW0 = 01: Row address width = 12 bits
  SAW1 and SAW0 = 01: Column address width = 9 bits

- Physical address

  A22 and A21: Bank addresses
  A20 to A9: Row addresses
  A8 to A0: Column addresses

- Address output from the MEMC

  A22 and A21: Bank addresses
  A11 to A0: Row address width (12 bits) and column address (9 bits)

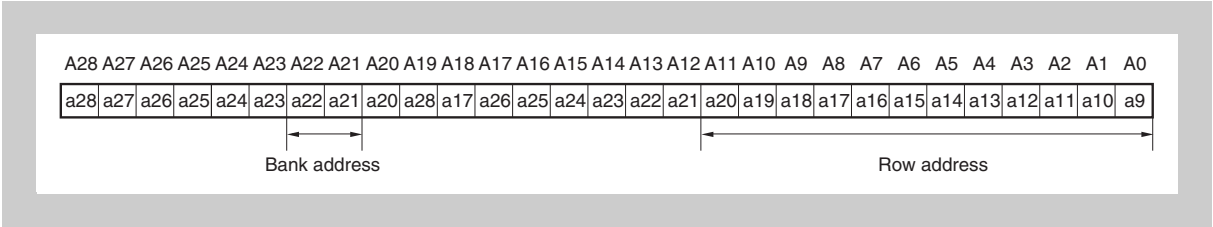
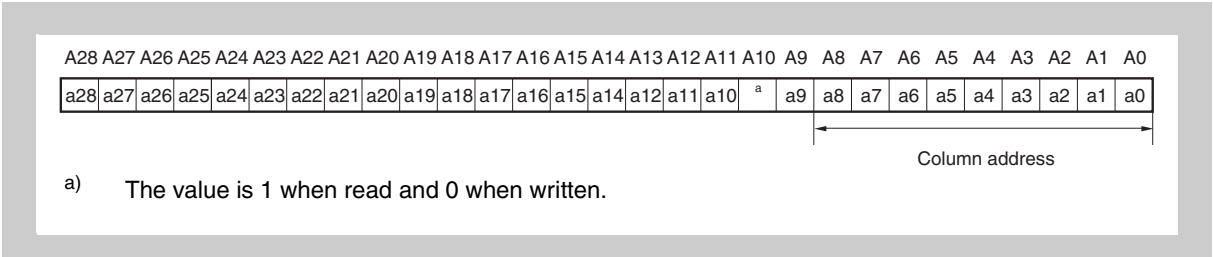


Figure 4-29 Row address and bank address output when an active command issued (on 8-bit bus)



a) The value is 1 when read and 0 when written.

Figure 4-30 Column address output when a read or write command is issued (on 8-bit bus)

- Connection of MEMC and SDRAM

  A22 and A21 (MEMC) to BA0 (A13) and BA1 (A12) (SDRAM)
  A11 to A0 (MEMC) to A11 to A0 (SDRAM)

(b) When the data bus width is 16 bits

The following shows an example of how to connect 512 Mb SDRAM (8M words × 16 bits × 4 banks) when the data bus width is 16 bits.

- SDCR register settings

  SSO1 and SSO0 = 01: Data bus width = 16 bits
  RAW1 and RAW0 = 10: Row address width = 13 bits
  SAW1 and SAW0 = 10: Column address width = 10 bits

- Physical address

  A25 and A24: Bank addresses
  A23 to A11: Row addresses
  A10 to A1: Column addresses

- Address output from the MEMC

  A25 and A24: Bank addresses
  A13 to A1: Row address width (13 bits) and column address (10 bits)

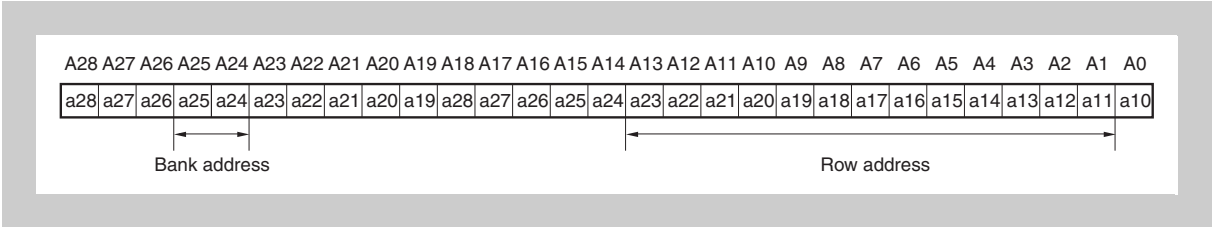
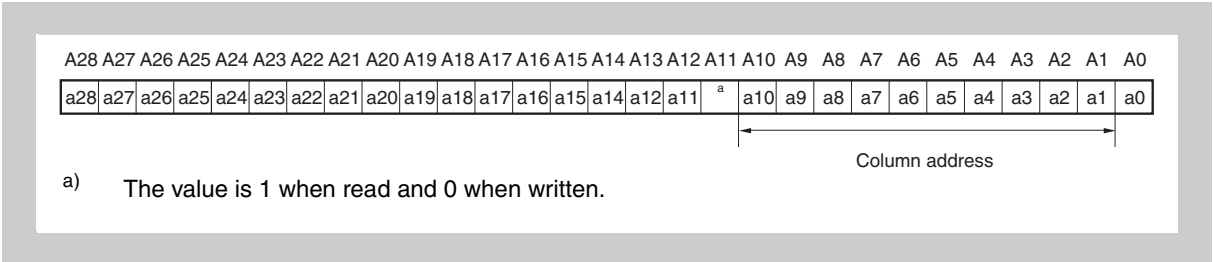


Figure 4-31 Row address and bank address output when an active command issued (on 16-bit bus)



a) The value is 1 when read and 0 when written.

Figure 4-32 Column address output when a read or write command is issued (on 16-bit bus)

- Connection of MEMC and SDRAM

  A25 and A24 (MEMC) to BA0 (A14) and BA1 (A13) (SDRAM)
  A13 to A1 (MEMC) to A12 to A0 (SDRAM)

(c) When the data bus width is 32 bits

The following shows an example of how to connect 256 Mb SDRAM (4M words × 16 bits × 4 banks × 2) when the data bus width is 32 bits.

- SDCR register settings

  SSO1 and SSO0 = 10: Data bus width = 32 bits
  RAW1 and RAW0 = 10: Row address width = 13 bits
  SAW1 and SAW0 = 01: Column address width = 9 bits

- Physical address

  A25 and A24: Bank addresses
  A23 to A11: Row addresses
  A10 to A2: Column addresses

- Address output from the MEMC

  A25 and A24: Bank addresses
  A14 to A2: Row address width (13 bits) and column address (9 bits)

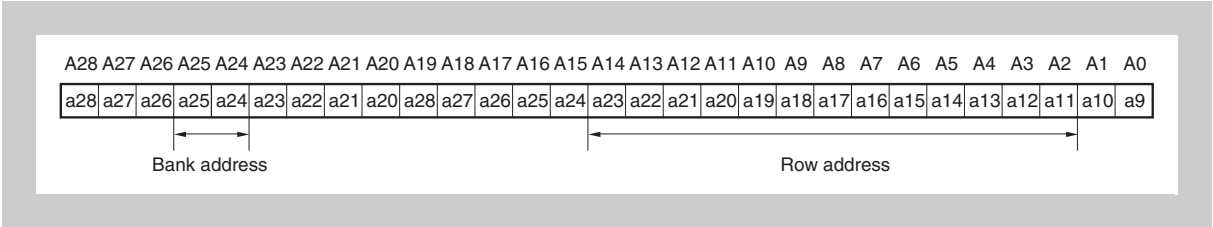
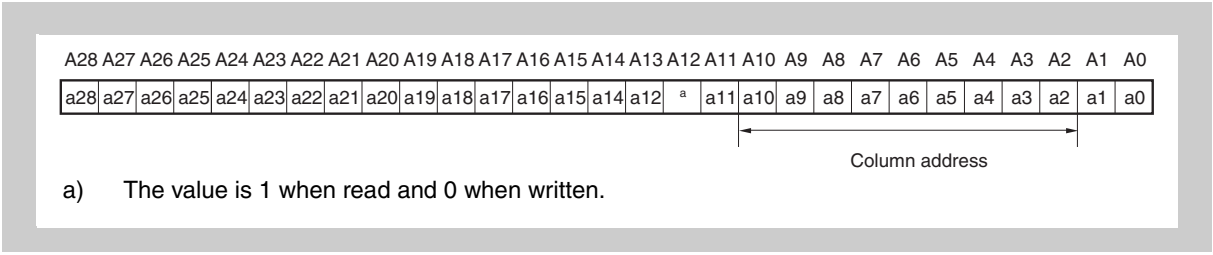


Figure 4-33 Row address and bank address output when an active command issued (on 32-bit bus)



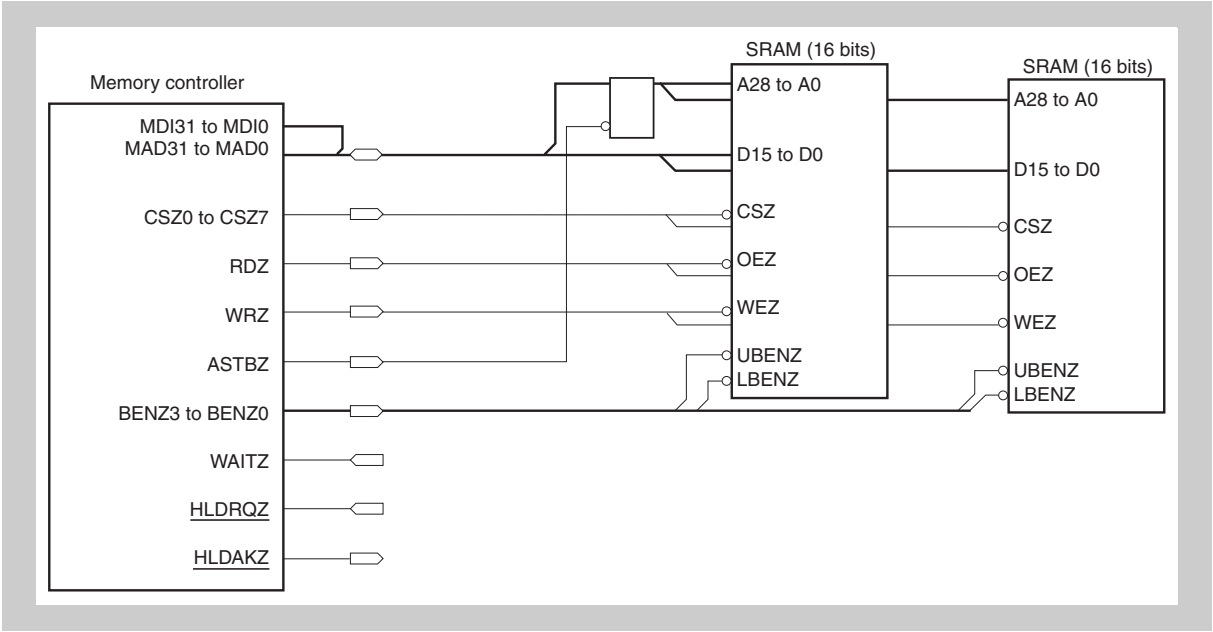
a) The value is 1 when read and 0 when written.

Figure 4-34 Column address output when a read or write command is issued (on 32-bit bus)

- Connection of MEMC and SDRAM

  A23 and A22 (MEMC) to BA0 (A14) and BA1 (A13) (SDRAM)
  A13 to A2 (MEMC) to A12 to A0 (SDRAM)

4.7.3 Multiplexed bus mode connection example



<R> **Figure 4-35 Multiplexed bus mode connection example**

4.8 Data Flow

The data transfer flow to external memory depends on conditions such as the data width, endianness, external bus width, and start address.

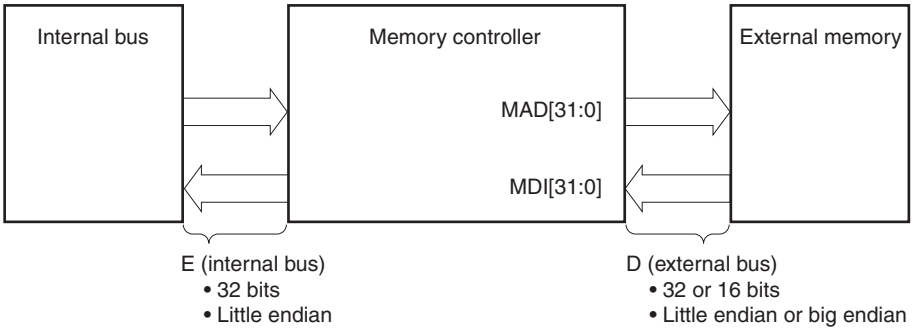
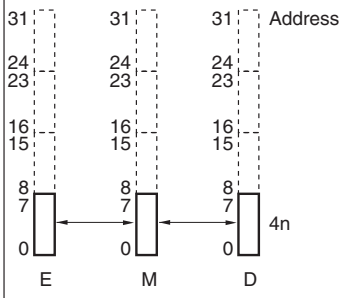
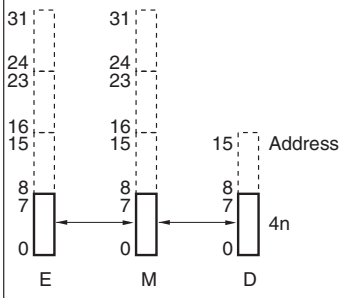
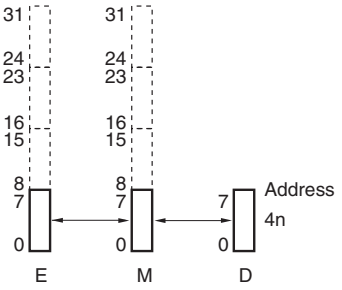
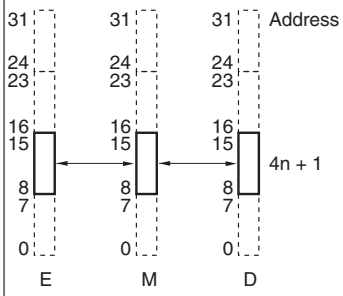
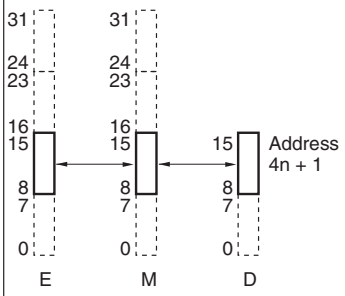
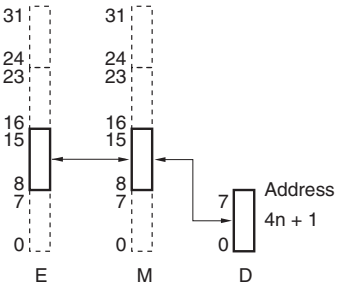
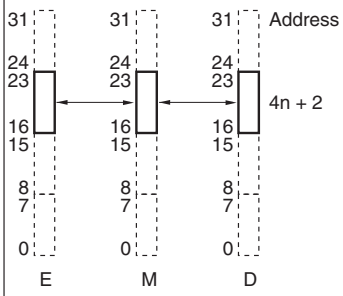
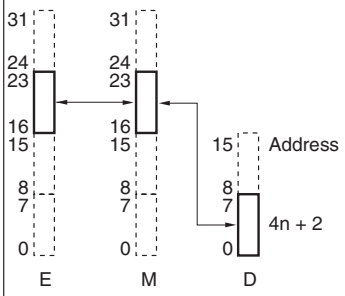
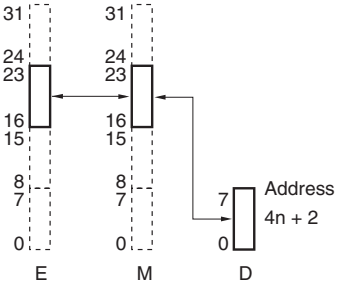
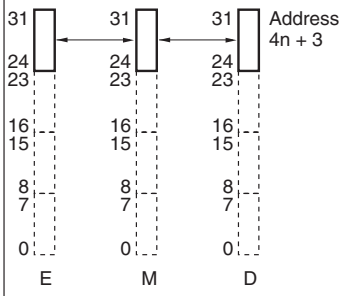
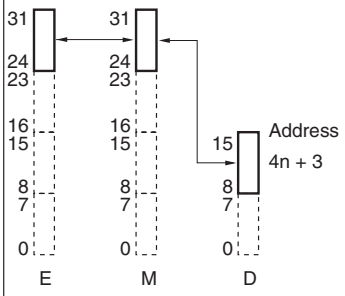
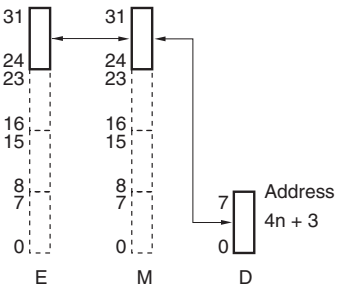


Figure 4-36 Flow of data through the internal bus, MEMC, and external bus

The data flows for various conditions are shown on the following pages.

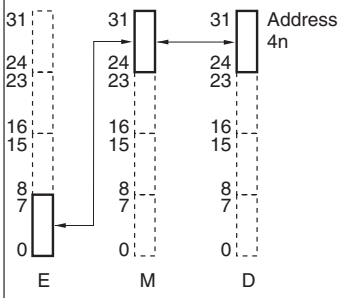
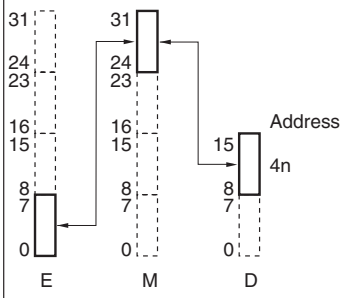
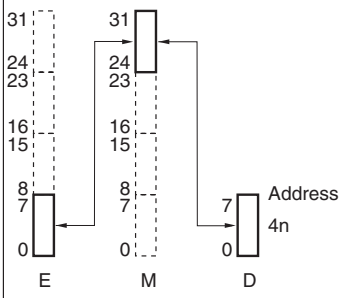
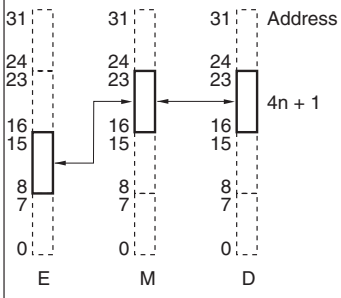
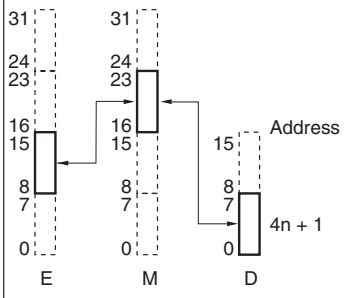
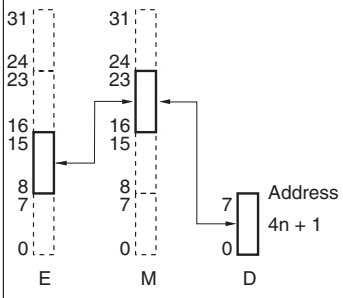
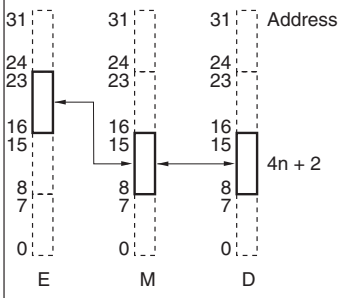
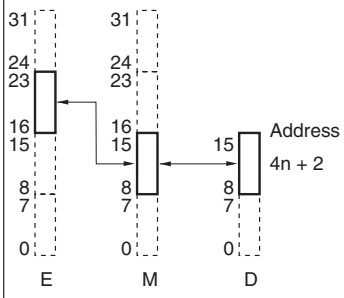
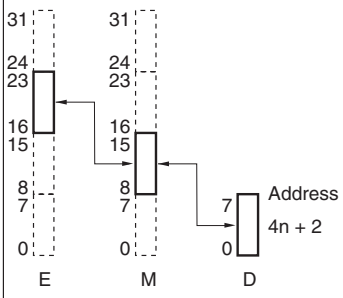
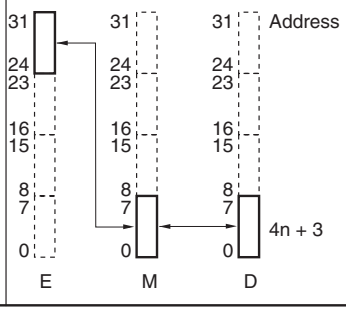
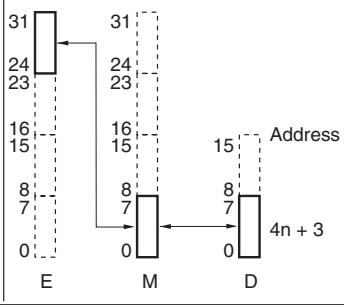
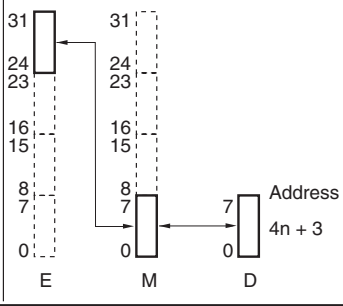
4.8.1 Data flow during byte access

Table 4-21 Data flow during byte access (little-endian)

Accessed address	Data transfer flow		
	External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n			
4n + 1			
4n + 2			
4n + 3			

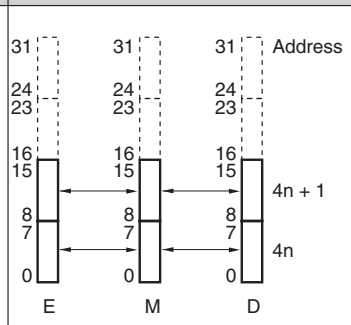
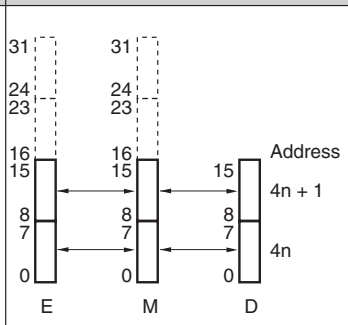
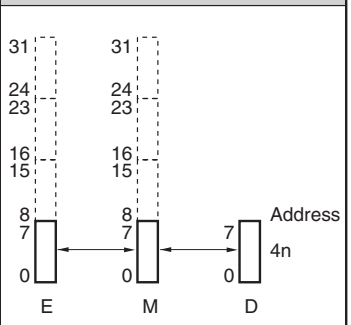
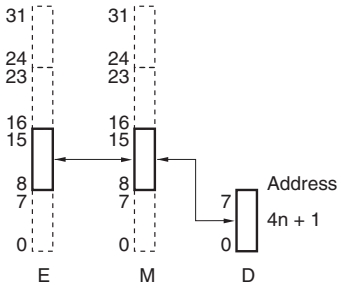
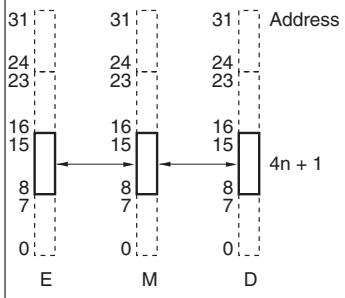
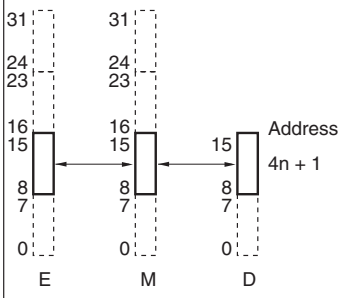
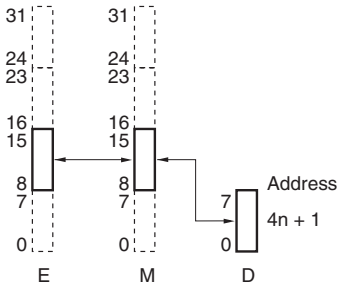
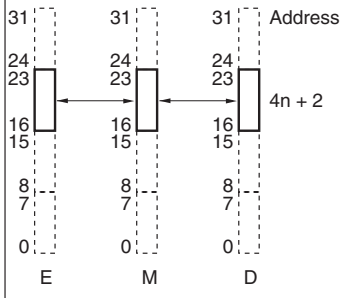
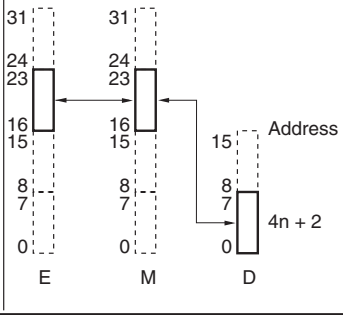
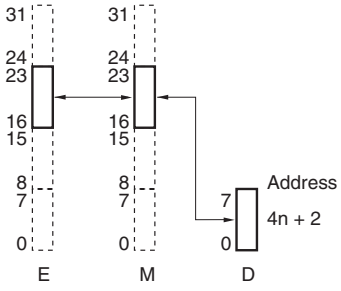
Note E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

Table 4-22 Data flow during byte access (big-endian)

Accessed address	Data transfer flow		
	External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n			
4n + 1			
4n + 2			
4n + 3			

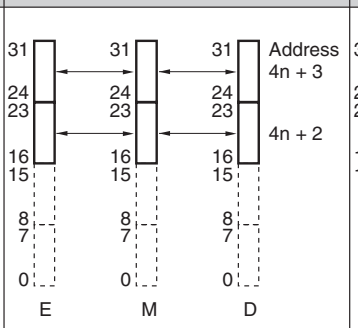
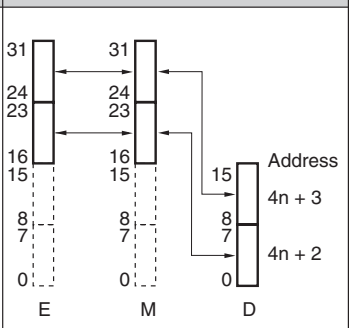
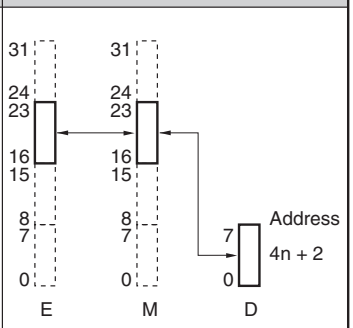
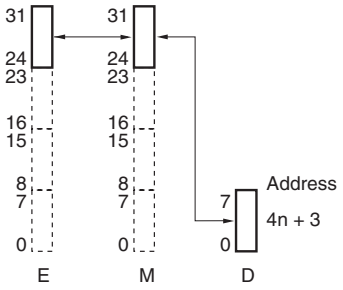
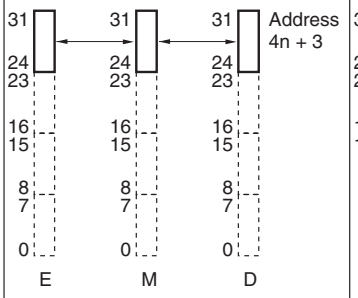
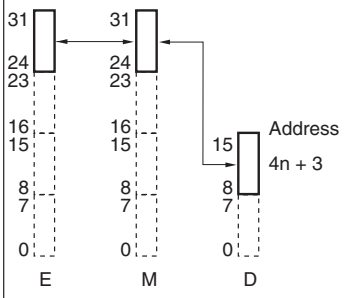
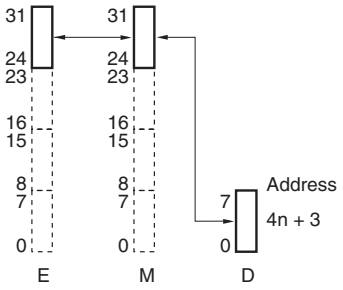
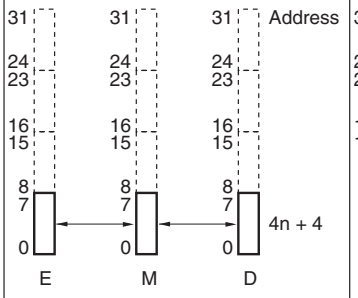
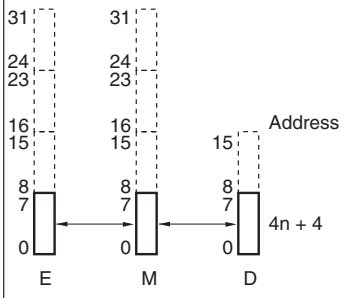
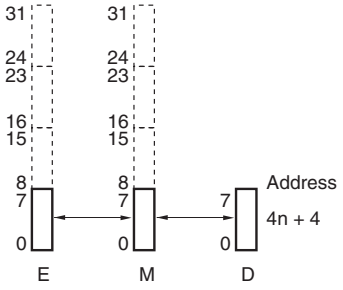
Note E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

Table 4-23 Data flow during halfword access (little-endian) (1/2)

Accessed address		Data transfer flow		
		External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n	1st			
	2nd	—	—	
4n + 1	1st			
	2nd			

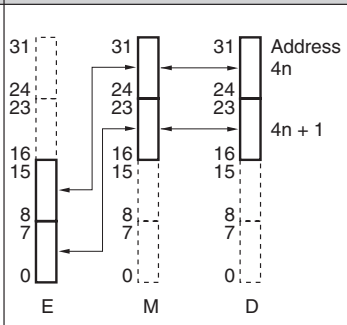
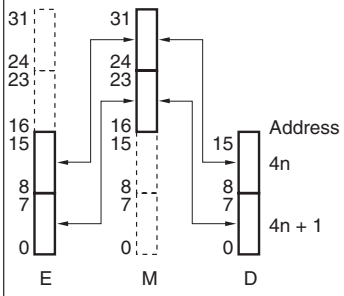
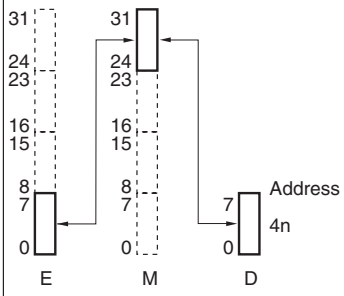
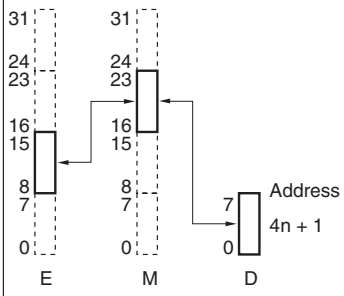
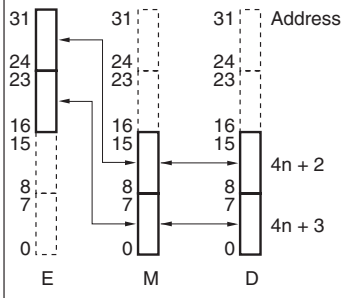
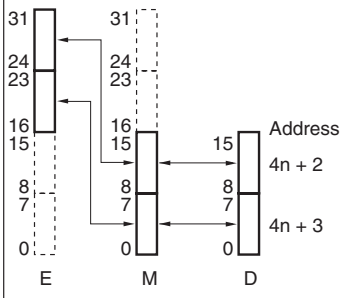
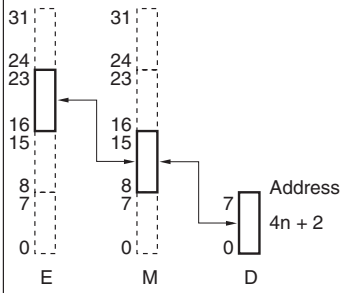
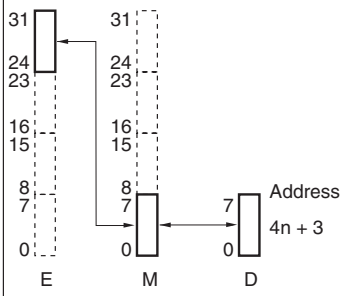
Note E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

Table 4-24 Data flow during halfword access (little-endian) (2/2)

Accessed address		Data transfer flow		
		External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n + 2	1st			
	2nd	—	—	
4n + 3	1st			
	2nd			

Note E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

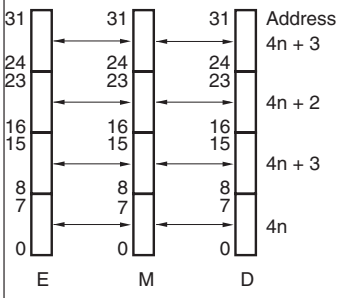
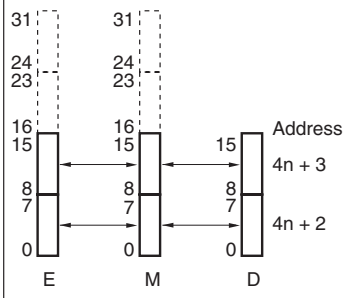
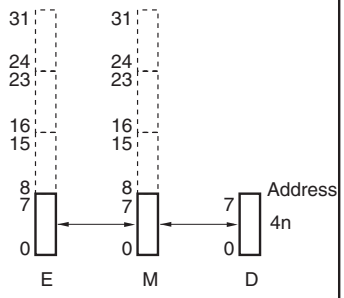
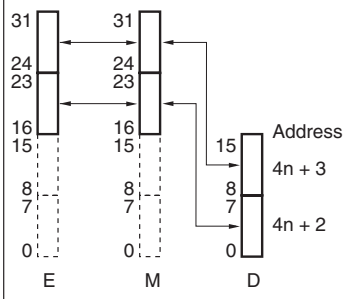
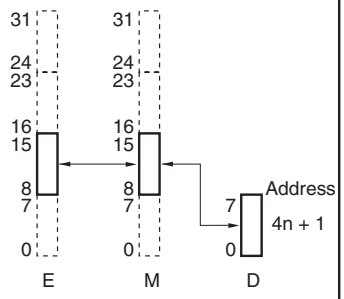
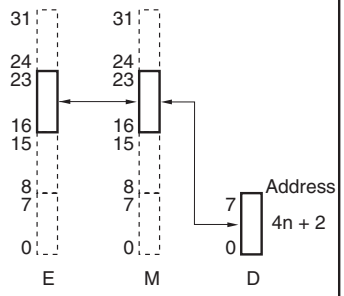
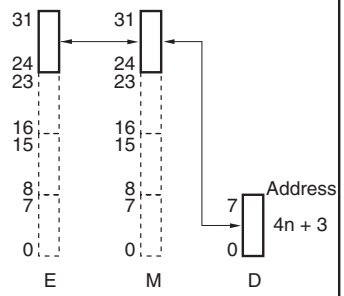
Table 4-25 Data flow during halfword access (big-endian)

Accessed address		Data transfer flow		
		External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n	1st			
	2nd	—	—	
4n + 2	1st			
	2nd	—	—	

Notes 1. E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

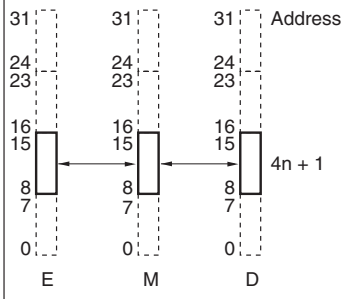
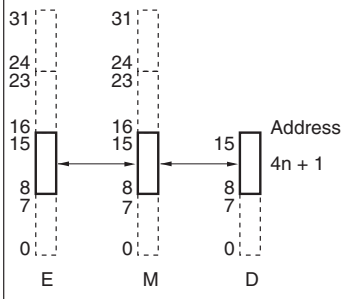
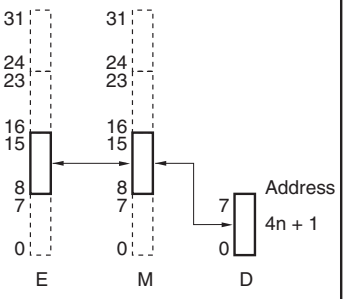
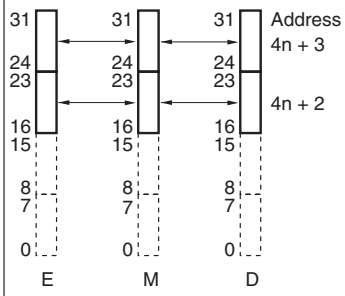
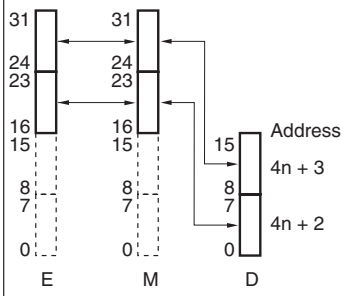
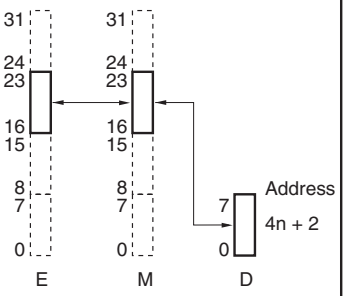
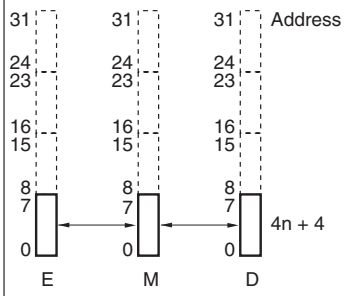
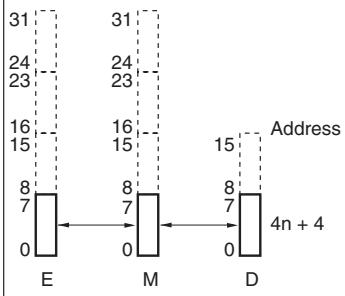
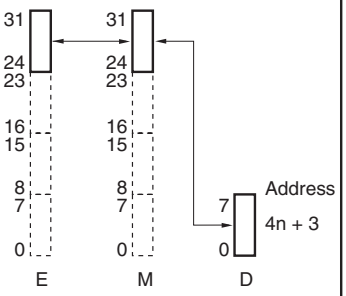
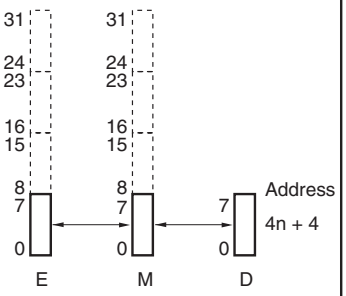
2. Accesses starting from address 4n+1 or 4n+3 are prohibited.

Table 4-26 Data flow during word access (little-endian) (1/4)

Accessed address		Data transfer flow		
		External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n	1st			
	2nd	—		
	3rd	—	—	
	4th	—	—	

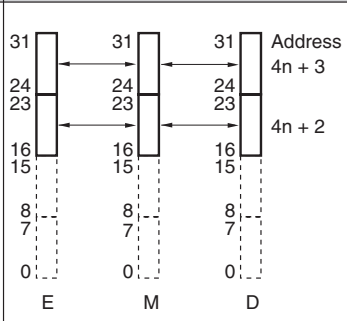
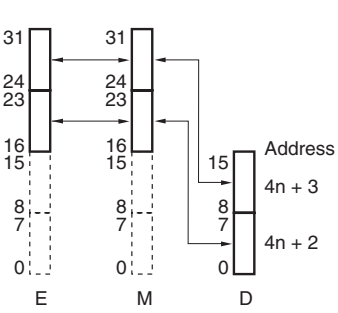
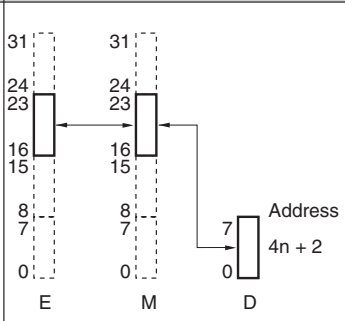
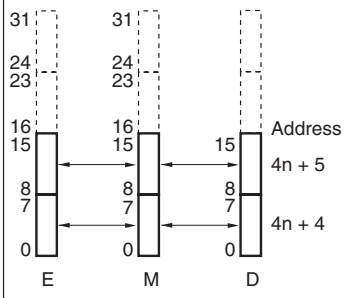
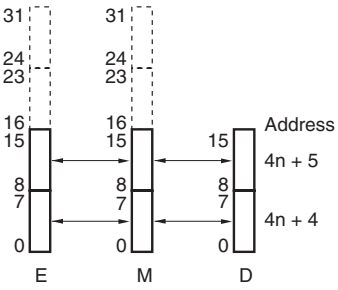
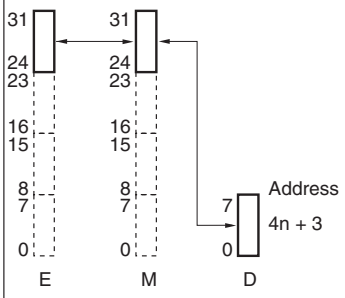
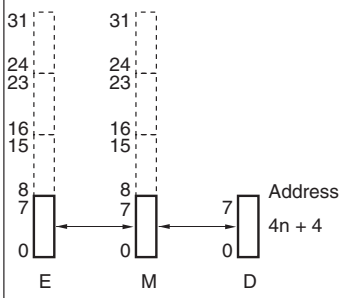
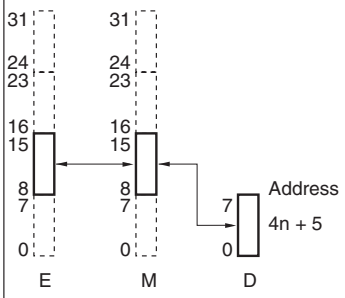
Note E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

Table 4-27 Data flow during word access (little-endian) (2/4)

Accessed address		Data transfer flow		
		External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n + 1	1st			
	2nd			
	3rd			
	4th	———	———	

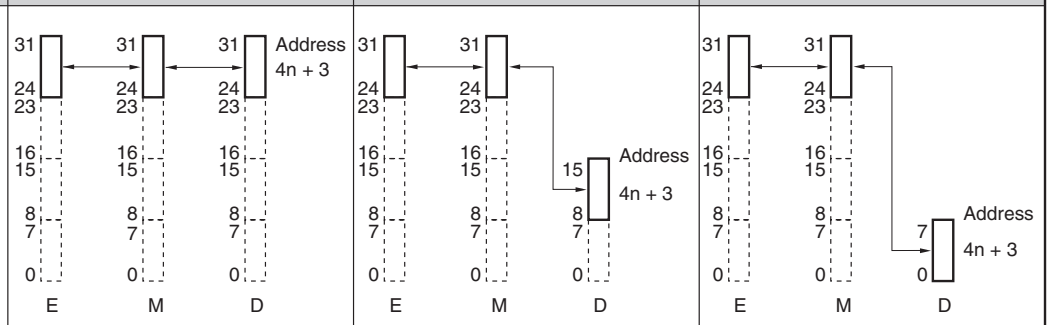
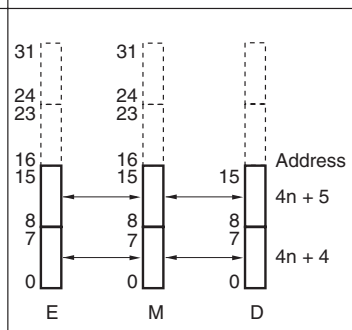
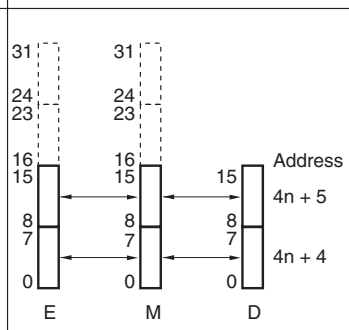
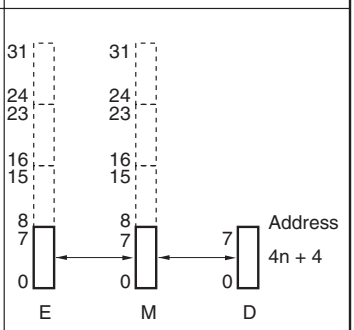
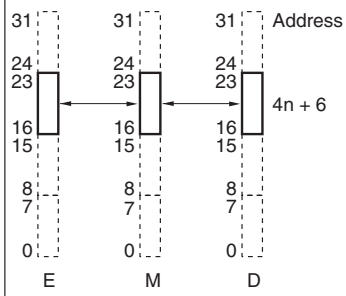
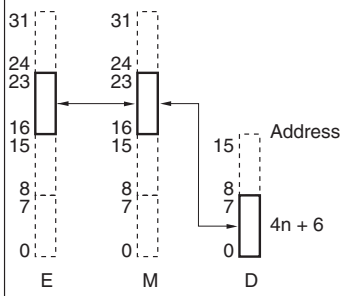
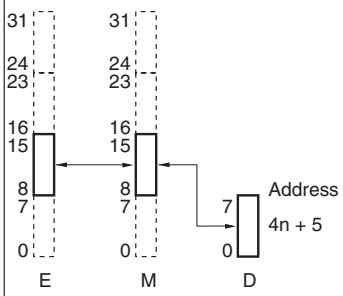
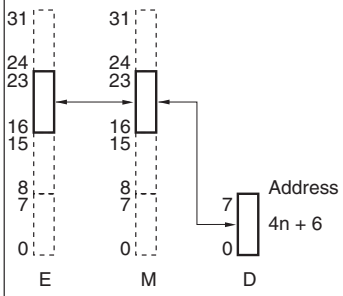
Note E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

Table 4-28 Data flow during word access (little-endian) (3/4)

Accessed address		Data transfer flow		
		External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n + 2	1st			
	2nd			
	3rd	———	———	
	4th	———	———	

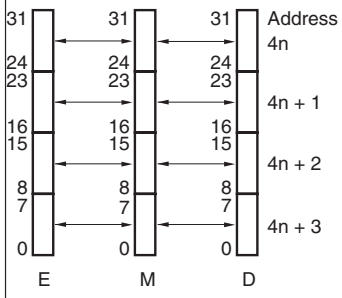
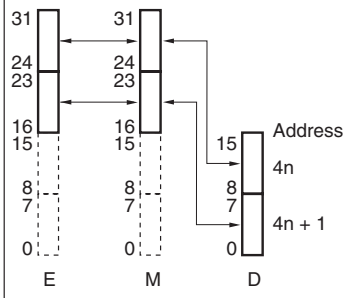
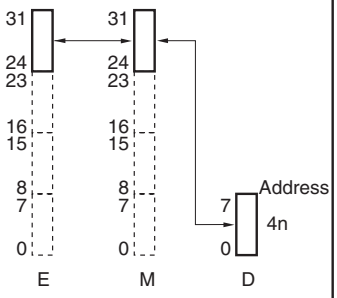
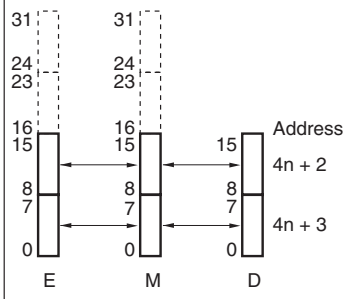
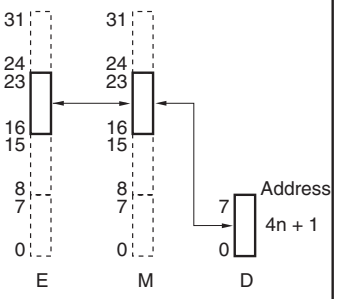
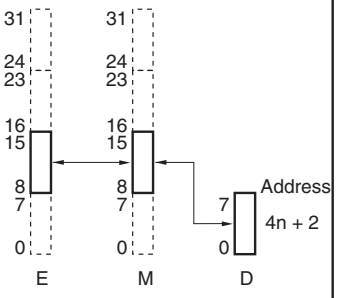
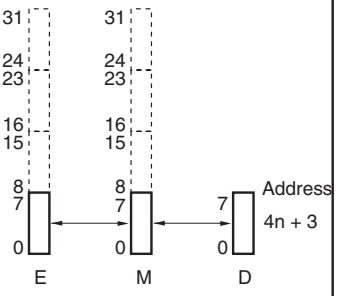
Note E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

Table 4-29 Data flow during word access (little-endian) (4/4)

Accessed address		Data transfer flow		
		External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n + 3	1st			
	2nd			
	3rd			
	4th	———	———	

Note E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

Table 4-30 Data flow during word access (big endian)

Accessed address		Data transfer flow		
		External data bus: 32 bits	External data bus: 16 bits	External data bus: 8 bits
4n	1st			
	2nd	—		
	3rd	—	—	
	4th	—	—	

Notes 1. E: Internal bus
M: MEMC data buffer
D: External data bus
n = 0, 1, 2, 3, …

2. Accesses starting from address 4n+1, 4n+2, or 4n+3 are prohibited.

Chapter 5 Interrupt Functions

5.1 Features

The phenomenon of forcing a branch operation from a currently running program to another program, due to a specific cause, is called an exception. This microcontroller supports the following types of exceptions.

For details about exceptions, see *V850E2M Architecture User's Manual*.

Table 5-1 Exception cause list

Name	Symbol	Cause group	Priority	Exception level
CPU initialization	RESET	Reset input	P1	–
FE level non-maskable interrupt[a]	FENMI	FENMI input	P2	FE
System error exception	SYSERR	SYSERR input (4 causes)	P3	FE
Peripheral device protection exception	PPI	Peripheral device protection violation	P4	FE
Timing monitoring exception	TSI	Timing monitoring violation	P5	FE
FE level maskable interrupt[a]	FEINT	FEINT input	P6	FE
Floating-point operation exception (imprecise)	FPI	FPU instruction	P7	EI
EI level maskable interrupt[a]	INT	Maskable interrupt input	P8	EI
Execution protection exception	MIP	Execution protection violation	P9	FE
Memory error exception	MEP	Instruction access error input	P10	FE
Data protection exception	MDP	Data protection violation	P11	FE
Floating-point operation exception (precise)	FPP	FPU instruction		EI
Coprocessor unusable exception	UCPOP	Coprocessor instruction		FE
Reserved instruction exception	RIEX	Reserved instruction		FE
FE level software exception	FETRAPEX	FETRAP instruction (vector = 1_H to F_H)		FE
EI level software exception	EITRAP0	TRAP0n instruction (vector = 00_H to $0F_H$)		EI
EI level software exception	EITRAP1	TRAP1n instruction (vector = 10_H to $1F_H$)		EI
System call exception	SYSCALLEX	SYSCALL instruction (vector = 00H to FFH)		EI

a) The description of these interrupt exceptions are subject to this chapter.

Priority Priority group P1 has the highest priority and P11 has the lowest.

Interrupt The following three types of exceptions in *Table 5-1 "Exception cause list"* are called interrupts, and are described in this chapter.

- FE level non-maskable interrupt (FENMI)
  FENMI is acknowledged and serviced immediately, even if another FE level interrupt (FENMI, FEINT) is in service.
  - It is serviced even when the NP bit of the CPU system register PSW is 1.
  - Resume disabled, recover disabled (for error report)

- FE level maskable interrupt (FEINT): One source
  FENMI is acknowledged and serviced if no other FE level interrupt (FENMI, FEINT) is in service.
  - It is serviced when the NP bit of the CPU system register PSW is 0. FEINT is masked when PSW.NP is 1.
  - Resume enabled, recover enabled
  - Highest priority interrupt (except FENMI)

- EI level maskable interrupt (EIINT)
  EIINT is acknowledged and serviced if no FE level interrupt (FENMI, FEINT) is in service
  - It is serviced when the NP bit of the CPU system register PSW is 0.
  - Resume enabled, recover enabled
  - Interrupt masking can be specified for each interrupt channel.
  - Up to 16 interrupt priority levels can be specified for each interrupt channel.
  - In this chapter, EIINT corresponding to interrupt channel n is written as EIINTn.

Resume Indicates whether the program can be resumed from the position where it was interrupted.

Recover Indicates whether the processor status (status of processor resources including general-purpose registers and system registers) at the time of a program interruption can be restored.

These interrupt sources are described below.

5.2 Interrupt Sources

5.2.1 Interrupt sources

(1) FE level non-maskable interrupt

Priority The FE level non-maskable interrupts have the priority P2.

Return PC The processor status cannot be recovered after this interrupt is serviced.

Control register FE level NMI control register

For details, see *(8) "FNC - FE level NMI control register"*.

Return instruction FERET instruction

Table 5-2 FE level non-maskable interrupt

Interrupt			Interrupt request		Unit	Priority	Exception code	Handler address 0000...
Symbol	Control register		Name	Source group				
	Name	Address FFFF...						
FENMI	FNC	$645C_H$	NMI	NMI input	Pin	P2	00020_H	0020_H
			WDTA0TNMI	Watchdog Timer 0 error NMI interrupt	WDTA0			
			WDTA1TNMI	Watchdog Timer 1 error NMI interrupt	WDTA1			

The FENMI interrupt source can be evaluated by using a dedicated flag register. For details, see *5.2.2 "Sharing of FE level non-maskable interrupt"*.

(2) FE level maskable interrupts

Priority The FE level maskable interrupts have the priority P6.

Return PC The program counter (PC) value, set after returning from any interrupt service routine by the FERET instruction is always the next address.

Control register FE level maskable interrupt control register

For details, see *(9) "FIC: FE level maskable interrupt control register"*.

Return instruction FERET instruction

Table 5-3 FE level non-maskable interrupt requests

Interrupt			Interrupt request		Unit	Priority	Exception code	Handler address 0000...
Symbol	Control register		Name	Source group				
	Name	Address FFFF...						
FEINT	FIC	$645E_H$	–	–	–	P6	00010_H	0010_H

(3) EI level maskable interrupts

Name The interrupt request signals, their assigned interrupt control registers, and the bits in these registers are named following the special rules below.

In the rules shown below, the interrupt request names are indicated by <*name*>.

- Interrupt request name: **INT**<*name*>
  The prefix "**INT**" is put in front of <*name*>.
- Interrupt request control register: **IC**<*name*>
  The prefix "**IC**" is put in front of <*name*>.
  The 16-bit registers **IC**<*name*> can also be accessed in 8- bit units.
  - Lower byte (bits [7:0]): **IC**<*name*>**L**
    The suffix "**L**" is appended to the register name **IC**<*name*>.
  - Upper byte (bits [15:8]): **IC**<*name*>**H**
    The suffix "**H**" is appended to the register name **IC**<*name*>
- Interrupt control register bit names: **RF**<*name*>, **MK**<*name*>, **P3**<*name*>, **P2**<*name*>, **P1**<*name*>, **P0**<*name*>
- The bit prefixes "**RF**", "**MK**", "**P3**", "**P2**", "**P1**", and "**P0**" are appended to the interrupt name <*name*>.
- Each interrupt request is assigned to a certain interrupt channel number n (n = 0 to 255).

In this chapter, the names of the interrupt requests, interrupt control registers, and control bits are described as follows. It is assumed that **INT**<*name*> is assigned to the interrupt channel number *n*.

- the interrupt request name: **EINT***n*
- the interrupt control register name: **EIC***n*
- the interrupt control bit names: **EIRF***n*, **EIMK***n*, **EIP3***n*, **EIP2***n*, **EIP1***n*, **EIP0***n*

Example The interrupt request of TAUA0 channel 2 (<*name*> = *TAUA0I2*) is named as:

INT*TAUA0I2*

The related interrupt control registers are names as:

IC*TAUA0I2*, **IC***TAUA0I2L*, **IC***TAUA0I2H*

The bits in this control register are named as:

RF*TAUA0I2*, **MK***TAUA0I2*, **P3***TAUA0I2*, **P2***TAUA0I2*, **P1***TAUA0I2*, **P0***TAUA0I2*

The interrupt channel n of **INT***TAUA0I2* is 22 (see *Table 5-4 "EI level maskable interrupts requests"*). Thus, in this chapter, the interrupt request name for this interrupt channel is:

EIINT*22*

The related interrupt control register name is:

EIC*22*

The bit names of the related interrupt control registers are:

EIRF*22*, **EIMK***22*, **EIP3***22*, **EIP2***22*, **EIP1***22*, **EIP0***22*

Table 5-4 "EI level maskable interrupts requests" lists the interrupt requests assigned to the V850E2/Sx4-H and their control registers, along with the interrupt channel number *n*.

Priority	The EI level maskable interrupts have the priority P8.
Return PC	The program counter (PC) value, set after returning from any interrupt service routine by the EIRET instruction is always the next address.
Control register	EI level maskable interrupt control register For details, see *(1) "EICn - EI level interrupt control registers (n = 0 to 255)"*.
Return instruction	EIRET instruction

(4) EI level maskable interrupts

Table 5-4 EI level maskable interrupts requests (1/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address 0000...	Supported by:		
Channel	Control register		Name	Source	Unit				V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
	Register name	Address FFFF...									
0	ICWDTA0	6000_H	INTWDTA0	WDTA0 75% interrupt	WDTA0	1	0080_H	0080_H	Yes	Yes	Yes
1	ICWDTA1	6002_H	INTWDTA1	WDTA1 75% interrupt	WDTA1	2	0090_H	0090_H	Yes	Yes	Yes
2	ICLVI	6004_H	INTLVI	LVI interrupt	LVI	3	$00A0_H$	$00A0_H$	Yes	Yes	Yes
3	R.F.U.	6006_H	R.F.U.			4	$00B0_H$	$00B0_H$	No	No	No
4	R.F.U.	6008_H	R.F.U.			5	$00C0_H$	$00C0_H$	No	No	No
5	R.F.U.	$600A_H$	R.F.U.			6	$00D0_H$	$00D0_H$	No	No	No
6	ICRTCA01S	$600C_H$	INTRTCA01S	RTC 1 Hz interrupt	RTCA0	7	$00E0_H$	$00E0_H$	Yes	Yes	Yes
7	ICRTCA0AL	$600E_H$	INTRTCA0AL	RTC alarm interrupt	RTCA0	8	$00F0_H$	$00F0_H$	Yes	Yes	Yes
8	ICRTCA0R	6010_H	INTRTCA0R	RTC fixed interval interrupt	RTCA0	9	0100_H	0100_H	Yes	Yes	Yes
9	ICP0	6012_H	INTP0	Pin input edge detection	Port	10	0110_H	0110_H	Yes	Yes	Yes
10	ICP1	6014_H	INTP1	Pin input edge detection	Port	11	0120_H	0120_H	Yes	Yes	Yes
11	ICP2	6016_H	INTP2	Pin input edge detection	Port	12	0130_H	0130_H	Yes	Yes	Yes
12	ICP3	6018_H	INTP3	Pin input edge detection	Port	13	0140_H	0140_H	Yes	Yes	Yes
13	ICP4	$601A_H$	INTP4	Pin input edge detection	Port	14	0150_H	0150_H	Yes	Yes	Yes
14	ICP5	$601C_H$	INTP5	Pin input edge detection	Port	15	0160_H	0160_H	Yes	Yes	Yes
15	ICP6	$601E_H$	INTP6	Pin input edge detection	Port	16	0170_H	0170_H	Yes	Yes	Yes
16	ICP7	6020_H	INTP7	Pin input edge detection	Port	17	0180_H	0180_H	Yes	Yes	Yes
17	ICP8	6022_H	INTP8	Pin input edge detection	Port	18	0190_H	0190_H	Yes	Yes	Yes
18	ICP9	6024_H	INTP9	Pin input edge detection	Port	19	$01A0_H$	$01A0_H$	Yes	Yes	Yes
19	ICP10	6026_H	INTP10	Pin input edge detection	Port	20	$01B0_H$	$01B0_H$	No	Yes	Yes
20	ICTAUA0I0	6028_H	INTTAUA0I0	Channel 0 interrupt	TAUA0	21	$01C0_H$	$01C0_H$	Yes	Yes	Yes
21	ICTAUA0I1	$602A_H$	INTTAUA0I1	Channel 1 interrupt	TAUA0	22	$01D0_H$	$01D0_H$	Yes	Yes	Yes
22	ICTAUA0I2	$602C_H$	INTTAUA0I2	Channel 2 interrupt	TAUA0	23	$01E0_H$	$01E0_H$	Yes	Yes	Yes
23	ICTAUA0I3	$602E_H$	INTTAUA0I3	Channel 3 interrupt	TAUA0	24	$01F0_H$	$01F0_H$	Yes	Yes	Yes
24	ICTAUA0I4	6030_H	INTTAUA0I4	Channel 4 interrupt	TAUA0	25	0200_H	0200_H	Yes	Yes	Yes
25	ICTAUA0I5	6032_H	INTTAUA0I5	Channel 5 interrupt	TAUA0	26	0210_H	0210_H	Yes	Yes	Yes
26	ICTAUA0I6	6034_H	INTTAUA0I6	Channel 6 interrupt	TAUA0	27	0220_H	0220_H	Yes	Yes	Yes
27	ICTAUA0I7	6036_H	INTTAUA0I7	Channel 7 interrupt	TAUA0	28	0230_H	0230_H	Yes	Yes	Yes
28	ICTAUA0I8	6038_H	INTTAUA0I8	Channel 8 interrupt	TAUA0	29	0240_H	0240_H	Yes	Yes	Yes
29	ICTAUA0I9	$603A_H$	INTTAUA0I9	Channel 9 interrupt	TAUA0	30	0250_H	0250_H	Yes	Yes	Yes
30	ICTAUA0I10	$603C_H$	INTTAUA0I10	Channel 10 interrupt	TAUA0	31	0260_H	0260_H	Yes	Yes	Yes
31	ICTAUA0I11	$603E_H$	INTTAUA0I11	Channel 11 interrupt	TAUA0	32	0270_H	0270_H	Yes	Yes	Yes
32	ICTAUA0I12	6040_H	INTTAUA0I12	Channel 12 interrupt	TAUA0	33	0280_H	0280_H	Yes	Yes	Yes
33	ICTAUA0I13	6042_H	INTTAUA0I13	Channel 13 interrupt	TAUA0	34	0290_H	0290_H	Yes	Yes	Yes
34	ICTAUA0I14	6044_H	INTTAUA0I14	Channel 14 interrupt	TAUA0	35	$02A0_H$	$02A0_H$	Yes	Yes	Yes
35	ICTAUA0I15	6046_H	INTTAUA0I15	Channel 15 interrupt	TAUA0	36	$02B0_H$	$02B0_H$	Yes	Yes	Yes

Table 5-4 EI level maskable interrupts requests (2/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address 0000...	Supported by:		
Channel	Control register		Name	Source	Unit				V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
	Register name	Address FFFF...									
36	ICTAUB2I0	6048_H	INTTAUB2I0	Channel 0 interrupt	TAUB2	37	$02C0_H$	$02C0_H$	No	No	Yes
37	ICTAUB2I1	$604A_H$	INTTAUB2I1	Channel 1 interrupt	TAUB2	38	$02D0_H$	$02D0_H$	No	No	Yes
38	ICTAUB2I2	$604C_H$	INTTAUB2I2	Channel 2 interrupt	TAUB2	39	$02E0_H$	$02E0_H$	No	No	Yes
39	ICTAUB2I3	$604E_H$	INTTAUB2I3	Channel 3 interrupt	TAUB2	40	$02F0_H$	$02F0_H$	No	No	Yes
40	ICTAUB2I4	6050_H	INTTAUB2I4	Channel 4 interrupt	TAUB2	41	0300_H	0300_H	No	No	Yes
41	ICTAUB2I5	6052_H	INTTAUB2I5	Channel 5 interrupt	TAUB2	42	0310_H	0310_H	No	No	Yes
42	ICTAUB2I6	6054_H	INTTAUB2I6	Channel 6 interrupt	TAUB2	43	0320_H	0320_H	No	No	Yes
43	ICTAUB2I7	6056_H	INTTAUB2I7	Channel 7 interrupt	TAUB2	44	0330_H	0330_H	No	No	Yes
44	ICTAUB2I8	6058_H	INTTAUB2I8	Channel 8 interrupt	TAUB2	45	0340_H	0340_H	No	No	Yes
45	ICTAUB2I9	$605A_H$	INTTAUB2I9	Channel 9 interrupt	TAUB2	46	0350_H	0350_H	No	No	Yes
46	ICTAUB2I10	$605C_H$	INTTAUB2I10	Channel 10 interrupt	TAUB2	47	0360_H	0360_H	No	No	Yes
47	ICTAUB2I11	$605E_H$	INTTAUB2I11	Channel 11 interrupt	TAUB2	48	0370_H	0370_H	No	No	Yes
48	ICTAUB2I12	6060_H	INTTAUB2I12	Channel 12 interrupt	TAUB2	49	0380_H	0380_H	No	No	Yes
49	ICTAUB2I13	6062_H	INTTAUB2I13	Channel 13 interrupt	TAUB2	50	0390_H	0390_H	No	No	Yes
50	ICTAUB2I14	6064_H	INTTAUB2I14	Channel 14 interrupt	TAUB2	51	$03A0_H$	$03A0_H$	No	No	Yes
51	ICTAUB2I15	6066_H	INTTAUB2I15	Channel 15 interrupt	TAUB2	52	$03B0_H$	$03B0_H$	No	No	Yes
52	R.F.U.	6068_H	R.F.U.			53	$03C0_H$	$03C0_H$	No	No	No
53	R.F.U.	$606A_H$	R.F.U.			54	$03D0_H$	$03D0_H$	No	No	No
54	ICIISA0IA	$606C_H$	INTIISA0IA	General interrupt	IISA0	55	$03E0_H$	$03E0_H$	Yes	Yes	Yes
55	ICIISA0ITXU	$606E_H$	INTIISA0ITXU	Transmission FIFO underrun interrupt	IISA0	56	$03F0_H$	$03F0_H$	Yes	Yes	Yes
56	ICIISA0ITXT	6070_H	INTIISA0ITXT	Transmission FIFO empty trigger level interrupt	IISA0	57	0400_H	0400_H	Yes	Yes	Yes
57	ICIISA0IRXO	6072_H	INTIISA0IRXO	Reception FIFO overrun interrupt	IISA0	58	0410_H	0410_H	Yes	Yes	Yes
58	ICIISA0IRXT	6074_H	INTIISA0IRXT	Reception FIFO full trigger level interrupt	IISA0	59	0420_H	0420_H	Yes	Yes	Yes
59	ICIISA0IFERR	6076_H	INTIISA0IFERR	Framing error interrupt	IISA0	60	0430_H	0430_H	Yes	Yes	Yes
60	ICIISA1IA	6078_H	INTIISA1IA	General interrupt	IISA1	61	0440_H	0440_H	Yes	Yes	Yes
61	ICIISA1ITXU	$607A_H$	INTIISA1ITXU	Transmission FIFO underrun interrupt	IISA1	62	0450_H	0450_H	Yes	Yes	Yes
62	ICIISA1ITXT	$607C_H$	INTIISA1ITXT	Transmission FIFO empty trigger level interrupt	IISA1	63	0460_H	0460_H	Yes	Yes	Yes
63	ICIISA1IRXO	$607E_H$	INTIISA1IRXO	Reception FIFO overrun interrupt	IISA1	64	0470_H	0470_H	Yes	Yes	Yes
64	ICIISA1IRXT	6080_H	INTIISA1IRXT	Reception FIFO full trigger level interrupt	IISA1	65	0480_H	0480_H	Yes	Yes	Yes
65	ICIISA1IFERR	6082_H	INTIISA1IFERR	Framing error interrupt	IISA1	66	0490_H	0490_H	Yes	Yes	Yes
66	ICIISA2IA	6084_H	INTIISA2IA	General interrupt	IISA2	67	$04A0_H$	$04A0_H$	Yes	Yes	Yes
67	ICIISA2ITXU	6086_H	INTIISA2ITXU	Transmission FIFO underrun interrupt	IISA2	68	$04B0_H$	$04B0_H$	Yes	Yes	Yes
68	ICIISA2ITXT	6088_H	INTIISA2ITXT	Transmission FIFO empty trigger level interrupt	IISA2	69	$04C0_H$	$04C0_H$	Yes	Yes	Yes
69	ICIISA2IRXO	$608A_H$	INTIISA2IRXO	Reception FIFO overrun interrupt	IISA2	70	$04D0_H$	$04D0_H$	Yes	Yes	Yes

Table 5-4 EI level maskable interrupts requests (3/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address 0000...	Supported by:		
Channel	Control register		Name	Source	Unit				V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
	Register name	Address FFFF...									
70	ICIISA2IRXT	$608C_H$	INTIISA2IRXT	Reception FIFO full trigger level interrupt	IISA2	71	$04E0_H$	$04E0_H$	Yes	Yes	Yes
71	ICIISA2IFERR	$608E_H$	INTIISA2IFERR	Framing error interrupt	IISA2	72	$04F0_H$	$04F0_H$	Yes	Yes	Yes
72	ICIISA3IA	6090_H	INTIISA3IA	General interrupt	IISA3	73	0500_H	0500_H	Yes	Yes	Yes
73	ICIISA3ITXU	6092_H	INTIISA3ITXU	Transmission FIFO underrun interrupt	IISA3	74	0510_H	0510_H	Yes	Yes	Yes
74	ICIISA3ITXT	6094_H	INTIISA3ITXT	Transmission FIFO empty trigger level interrupt	IISA3	75	0520_H	0520_H	Yes	Yes	Yes
75	ICIISA3IRXO	6096_H	INTIISA3IRXO	Reception FIFO overrun interrupt	IISA3	76	0530_H	0530_H	Yes	Yes	Yes
76	ICIISA3IRXT	6098_H	INTIISA3IRXT	Reception FIFO full trigger level interrupt	IISA3	77	0540_H	0540_H	Yes	Yes	Yes
77	ICIISA3IFERR	$609A_H$	INTIISA3IFERR	Framing error interrupt	IISA3	78	0550_H	0550_H	Yes	Yes	Yes
78	ICIISA4IA	$609C_H$	INTIISA4IA	General interrupt	IISA4	79	0560_H	0560_H	No	No	Yes
79	ICIISA4ITXU	$609E_H$	INTIISA4ITXU	Transmission FIFO underrun interrupt	IISA4	80	0570_H	0570_H	No	No	Yes
80	ICIISA4ITXT	$60A0_H$	INTIISA4ITXT	Transmission FIFO empty trigger level interrupt	IISA4	81	0580_H	0580_H	No	No	Yes
81	ICIISA4IRXO	$60A2_H$	INTIISA4IRXO	Reception FIFO overrun interrupt	IISA4	82	0590_H	0590_H	No	No	Yes
82	ICIISA4IRXT	$60A4_H$	INTIISA4IRXT	Reception FIFO full trigger level interrupt	IISA4	83	$05A0_H$	$05A0_H$	No	No	Yes
83	ICIISA4IFERR	$60A6_H$	INTIISA4IFERR	Framing error interrupt	IISA4	84	$05B0_H$	$05B0_H$	No	No	Yes
84	ICIISA5IA	$60A8_H$	INTIISA5IA	General interrupt	IISA5	85	$05C0_H$	$05C0_H$	No	No	Yes
85	ICIISA5ITXU	$60AA_H$	INTIISA5ITXU	Transmission FIFO underrun interrupt	IISA5	86	$05D0_H$	$05D0_H$	No	No	Yes
86	ICIISA5ITXT	$60AC_H$	INTIISA5ITXT	Transmission FIFO empty trigger level interrupt	IISA5	87	$05E0_H$	$05E0_H$	No	No	Yes
87	ICIISA5IRXO	$60AE_H$	INTIISA5IRXO	Reception FIFO overrun interrupt	IISA5	88	$05F0_H$	$05F0_H$	No	No	Yes
88	ICIISA5IRXT	$60B0_H$	INTIISA5IRXT	Reception FIFO full trigger level interrupt	IISA5	89	0600_H	0600_H	No	No	Yes
89	ICIISA5IFERR	$60B2_H$	INTIISA5IFERR	Framing error interrupt	IISA5	90	0610_H	0610_H	No	No	Yes
90	ICPM0IC	$60B4_H$	INTPM0INT	General interrupt	PCM0	91	0620_H	0620_H	Yes	Yes	Yes
91	ICPM0RXREN	$60B6_H$	INTPM0RXREN	Reception data FIFO interrupt	PCM0	92	0630_H	0630_H	Yes	Yes	Yes
92	ICPM0RXORE	$60B8_H$	INTPM0RXORE	Reception overrun error interrupt	PCM0	93	0640_H	0640_H	Yes	Yes	Yes
93	ICPM0RXURE	$60BA_H$	INTPM0RXURE	Reception underrun error interrupt	PCM0	94	0650_H	0650_H	Yes	Yes	Yes
94	ICPM0RXFRE	$60BC_H$	INTPM0RXFRE	Reception frame synchronization error interrupt	PCM0	95	0660_H	0660_H	Yes	Yes	Yes
95	ICPM0TXWEN	$60BE_H$	INTPM0TXWEN	Transmission data FIFO interrupt	PCM0	96	0670_H	0670_H	Yes	Yes	Yes
96	ICPM0TXORE	$60C0_H$	INTPM0TXORE	Transmission overrun error interrupt	PCM0	97	0680_H	0680_H	Yes	Yes	Yes
97	ICPM0TXURE	$60C2_H$	INTPM0TXURE	Transmission underrun error interrupt	PCM0	98	0690_H	0690_H	Yes	Yes	Yes
98	ICPM0TXFRE	$60C4_H$	INTPM0TXFRE	Transmission frame synchronization error interrupt	PCM0	99	$06A0_H$	$06A0_H$	Yes	Yes	Yes
99	R.F.U.	$60C6_H$	R.F.U.			100	$06B0_H$	$06B0_H$	No	No	No
100	ICADCA0ERR	$60C8_H$	INTADCA0ERR	Error interrupt	ADCA0	101	$06C0_H$	$06C0_H$	Yes	Yes	Yes

Table 5-4 EI level maskable interrupts requests (4/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address 0000...	Supported by:		
Channel	Control register: Register name	Control register: Address FFFF...	Name	Source	Unit				V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
101	ICADCA0I0	$60CA_H$	INTADCA0I0	End of conversion on CG0	ADCA0	102	$06D0_H$	$06D0_H$	Yes	Yes	Yes
102	ICADCA0I1	$60CC_H$	INTADCA0I1	End of conversion on CG1	ADCA0	103	$06E0_H$	$06E0_H$	Yes	Yes	Yes
103	ICADCA0I2	$60CE_H$	INTADCA0I2	End of conversion on CG2	ADCA0	104	$06F0_H$	$06F0_H$	Yes	Yes	Yes
104	ICADCA0LLT	$60D0_H$	INTADCA0LLT	Conversion interrupt	ADCA0	105	0700_H	0700_H	Yes	Yes	Yes
105	ICFCNWUP	$60D2_H$	INTFCNWUP	Wake-up interrupt	FCN[1:0]	106	0710_H	0710_H	Yes	Yes	Yes
106	ICFCN0ERR	$60D4_H$	INTFCN0ERR	Error interrupt	FCN0	107	0720_H	0720_H	Yes	Yes	Yes
107	ICFCN0REC	$60D6_H$	INTFCN0REC	Reception interrupt	FCN0	108	0730_H	0730_H	Yes	Yes	Yes
108	ICFCN0TRX	$60D8_H$	INTFCN0TRX	Transmission interrupt	FCN0	109	0740_H	0740_H	Yes	Yes	Yes
109	ICCSIG0IRE	$60DA_H$	INTCSIG0IRE	Reception error interrupt	CSIG0	110	0750_H	0750_H	Yes	Yes	Yes
110	ICCSIG0IR	$60DC_H$	INTCSIG0IR	Reception status interrupt	CSIG0	111	0760_H	0760_H	Yes	Yes	Yes
111	ICCSIG0IC	$60DE_H$	INTCSIG0IC	Communication status interrupt	CSIG0	112	0770_H	0770_H	Yes	Yes	Yes
112	ICLMA0IS	$60E0_H$	INTLMA0IS[a]	Status interrupt	LMA0	113	0780_H	0780_H	Yes	Yes	Yes
113	ICLMA0IR	$60E2_H$	INTLMA0IR[a]	Reception completion interrupt	LMA0	114	0790_H	0790_H	Yes	Yes	Yes
114	ICLMA0IT	$60E4_H$	INTLMA0IT[a]	Transfer interrupt	LMA0	115	$07A0_H$	$07A0_H$	Yes	Yes	Yes
115	ICLMA1IS	$60E6_H$	INTLMA1IS[a]	Status interrupt	LMA1	116	$07B0_H$	$07B0_H$	Yes	Yes	Yes
116	ICLMA1IR	$60E8_H$	INTLMA1IR[a]	Reception completion interrupt	LMA1	117	$07C0_H$	$07C0_H$	Yes	Yes	Yes
117	ICLMA1IT	$60EA_H$	INTLMA1IT[a]	Transfer interrupt	LMA1	118	$07D0_H$	$07D0_H$	Yes	Yes	Yes
118	R.F.U.	$60EC_H$	R.F.U.			119	$07E0_H$	$07E0_H$	No	No	No
119	ICDMA0	$60EE_H$	INTDMA0	DMA channel 0 transfer completion (or INTCT0[b] count match interrupt)	DMA	120	$07F0_H$	$07F0_H$	Yes	Yes	Yes
	ICCT0		INTCT0		DMA				Yes	Yes	Yes
120	ICDMA1	$60F0_H$	INTDMA1	DMA channel 1 transfer completion (or INTCT1[b] count match interrupt)	DMA	121	0800_H	0800_H	Yes	Yes	Yes
	ICCT1		INTCT1		DMA				Yes	Yes	Yes
121	ICDMA2	$60F2_H$	INTDMA2	DMA channel 2 transfer completion (or INTCT2[b] count match interrupt)	DMA	122	0810_H	0810_H	Yes	Yes	Yes
	ICCT2		INTCT2		DMA				Yes	Yes	Yes
122	ICDMA3	$60F4_H$	INTDMA3	DMA channel 3 transfer completion (or INTCT3[b] count match interrupt)	DMA	123	0820_H	0820_H	Yes	Yes	Yes
	ICCT3		INTCT3		DMA				Yes	Yes	Yes
123	ICDMA4	$60F6_H$	INTDMA4	DMA channel 4 transfer completion (or INTCT4[b] count match interrupt)	DMA	124	0830_H	0830_H	Yes	Yes	Yes
	ICCT4		INTCT4		DMA				Yes	Yes	Yes
124	ICDMA5	$60F8_H$	INTDMA5	DMA channel 5 transfer completion (or INTCT5[b] count match interrupt)	DMA	125	0840_H	0840_H	Yes	Yes	Yes
	ICCT5		INTCT5		DMA				Yes	Yes	Yes
125	ICDMA6	$60FA_H$	INTDMA6	DMA channel 6 transfer completion (or INTCT6[b] count match interrupt)	DMA	126	0850_H	0850_H	Yes	Yes	Yes
	ICCT6		INTCT6		DMA				Yes	Yes	Yes
126	ICDMA7	$60FC_H$	INTDMA7	DMA channel 7 transfer completion (or INTCT7[b] count match interrupt)	DMA	127	0860_H	0860_H	Yes	Yes	Yes
	ICCT7		INTCT7		DMA				Yes	Yes	Yes
127	R.F.U.	$60FE_H$	R.F.U.			128	0870_H	0870_H	No	No	No
128	ICIICB0IS	6100_H	INTIICB0IS	Status interrupt	IICB0	129	0880_H	0880_H	Yes	Yes	Yes
129	ICIICB0IA	6102_H	INTIICB0IA	Data transmission/reception interrupt	IICB0	130	0890_H	0890_H	Yes	Yes	Yes

Table 5-4 EI level maskable interrupts requests (5/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address 0000...	Supported by:		
Channel	Control register		Name	Source	Unit				V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
	Register name	Address FFFF...									
130	ICIICB1IS	6104_H	INTIICB1IS	Status interrupt	IICB1	131	$08A0_H$	$08A0_H$	Yes	Yes	Yes
131	ICIICB1IA	6106_H	INTIICB1IA	Data transmission/reception interrupt	IICB1	132	$08B0_H$	$08B0_H$	Yes	Yes	Yes
132	ICFCN1ERR	6108_H	INTFCN1ERR	Error interrupt	FCN1	133	$08C0_H$	$08C0_H$	No	Yes	Yes
133	ICFCN1REC	$610A_H$	INTFCN1REC	Reception interrupt	FCN1	134	$08D0_H$	$08D0_H$	No	Yes	Yes
134	ICFCN1TRX	$610C_H$	INTFCN1TRX	Transmission interrupt	FCN1	135	$08E0_H$	$08E0_H$	No	Yes	Yes
135	ICTAUJ0I0	$610E_H$	INTTAUJ0I0	Channel 0 interrupt	TAUJ0	136	$08F0_H$	$08F0_H$	Yes	Yes	Yes
136	ICTAUJ0I1	6110_H	INTTAUJ0I1	Channel 1 interrupt	TAUJ0	137	0900_H	0900_H	Yes	Yes	Yes
137	ICTAUJ0I2	6112_H	INTTAUJ0I2	Channel 2 interrupt	TAUJ0	138	0910_H	0910_H	Yes	Yes	Yes
138	ICTAUJ0I3	6114_H	INTTAUJ0I3	Channel 3 interrupt	TAUJ0	139	0920_H	0920_H	Yes	Yes	Yes
139	R.F.U.	6116_H	R.F.U.			140	0930_H	0930_H	No	No	No
140	R.F.U.	6118_H	R.F.U.			141	0940_H	0940_H	No	No	No
141	R.F.U.	$611A_H$	R.F.U.			142	0950_H	0950_H	No	No	No
142	R.F.U.	$611C_H$	R.F.U.			143	0960_H	0960_H	No	No	No
143	R.F.U.	$611E_H$	R.F.U.			144	0970_H	0970_H	No	No	No
144	R.F.U.	6120_H	R.F.U.			145	0980_H	0980_H	No	No	No
145	R.F.U.	6122_H	R.F.U.			146	0990_H	0990_H	No	No	No
146	R.F.U.	6124_H	R.F.U.			147	$09A0_H$	$09A0_H$	No	No	No
147	ICOSTM0	6126_H	INTOSTM0	OSTM0 interrupt	OSTM0	148	$09B0_H$	$09B0_H$	Yes	Yes	Yes
148	R.F.U.	6128_H	R.F.U.			149	$09C0_H$	$09C0_H$	No	No	No
149	R.F.U.	$612A_H$	R.F.U.			150	$09D0_H$	$09D0_H$	No	No	No
150	R.F.U.	$612C_H$	R.F.U.			151	$09E0_H$	$09E0_H$	No	No	No
151	R.F.U.	$612E_H$	R.F.U.			152	$09F0_H$	$09F0_H$	No	No	No
152	R.F.U.	6130_H	R.F.U.			153	$0A00_H$	$0A00_H$	No	No	No
153	R.F.U.	6132_H	R.F.U.			154	$0A10_H$	$0A10_H$	No	No	No
154	ICIICB2IS	6134_H	INTIICB2IS	Status interrupt	IICB2	155	$0A20_H$	$0A20_H$	Yes	Yes	Yes
155	ICIICB2IA	6136_H	INTIICB2IA	Data transmission/reception interrupt	IICB2	156	$0A30_H$	$0A30_H$	Yes	Yes	Yes
156	ICIICB3IS	6138_H	INTIICB3IS	Status interrupt	IICB3	157	$0A40_H$	$0A40_H$	Yes	Yes	Yes
157	ICIICB3IA	$613A_H$	INTIICB3IA	Data transmission/reception interrupt	IICB3	158	$0A50_H$	$0A50_H$	Yes	Yes	Yes
158	R.F.U.	$613C_H$	R.F.U.			159	$0A60_H$	$0A60_H$	No	No	No
159	R.F.U.	$613E_H$	R.F.U.			160	$0A70_H$	$0A70_H$	No	No	No
160	ICETHA0SCTXTCH	6140_H	INTETHA0SCTXTCH	Ethernet transmission data calculation completion interrupt	ETHA0	161	$0A80_H$	$0A80_H$	No	No	Yes
161	ICETHA0SCRXTCH	6142_H	INTETHA0SCRXTCH	Ethernet transmission checksum interrupt	ETHA0	162	$0A90_H$	$0A90_H$	No	No	Yes
162	R.F.U.	6144_H	R.F.U.			163	$0AA0_H$	$0AA0_H$	No	No	No
163	ICCSIH0IC	6146_H	INTCSIH0IC	Communication status interrupt	CSIH0	164	$0AB0_H$	$0AB0_H$	Yes	Yes	Yes
164	ICCSIH0IJC	6148_H	INTCSIH0IJC	Job completion interrupt	CSIH0	165	$0AC0_H$	$0AC0_H$	Yes	Yes	Yes
165	R.F.U.	$614A_H$	R.F.U.			166	$0AD0_H$	$0AD0_H$	No	No	No

Table 5-4 EI level maskable interrupts requests (6/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address 0000...	Supported by:		
Channel	Control register: Register name	Control register: Address FFFF...	Name	Source	Unit				V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
166	R.F.U.	$614C_H$	R.F.U.			167	$0AE0_H$	$0AE0_H$	No	No	No
167	R.F.U.	$614E_H$	R.F.U.			168	$0AF0_H$	$0AF0_H$	No	No	No
168	ICCSIH0IRE	6150_H	INTCSIH0IRE	Reception error interrupt	CSIH0	169	$0B00_H$	$0B00_H$	Yes	Yes	Yes
169	ICCSIH0IR	6152_H	INTCSIH0IR	Reception status interrupt	CSIH0	170	$0B10_H$	$0B10_H$	Yes	Yes	Yes
170	ICCSIG4IRE	6154_H	INTCSIG4IRE	Reception error interrupt	CSIG4	171	$0B20_H$	$0B20_H$	Yes	Yes	Yes
171	ICCSIG4IR	6156_H	INTCSIG4IR	Reception status interrupt	CSIG4	172	$0B30_H$	$0B30_H$	Yes	Yes	Yes
172	ICCSIG4IC	6158_H	INTCSIG4IC	Communication status interrupt	CSIG4	173	$0B40_H$	$0B40_H$	Yes	Yes	Yes
173	R.F.U.	$615A_H$	R.F.U.			174	$0B50_H$	$0B50_H$	No	No	No
174	R.F.U.	$615C_H$	R.F.U.			175	$0B60_H$	$0B60_H$	No	No	No
175	R.F.U.	$615E_H$	R.F.U.			176	$0B70_H$	$0B70_H$	No	No	No
176	ICIEBB0D	6160_H	INTIEBB0D	Data interrupt	IEBB0	177	$0B80_H$	$0B80_H$	Yes	Yes	Yes
177	ICIEBB0V	6162_H	INTIEBB0V	Vectored interrupt	IEBB0	178	$0B90_H$	$0B90_H$	Yes	Yes	Yes
178	ICIEBB0ERR	6164_H	INTIEBB0ERR	Error interrupt	IEBB0	179	$0BA0_H$	$0BA0_H$	Yes	Yes	Yes
179	ICIEBB0STA	6166_H	INTIEBB0STA	Status interrupt	IEBB0	180	$0BB0_H$	$0BB0_H$	Yes	Yes	Yes
180	ICCSIH1IRE	6168_H	INTCSIH1IRE	Reception error interrupt	CSIH1	181	$0BC0_H$	$0BC0_H$	Yes	Yes	Yes
181	ICCSIH1IR	$616A_H$	INTCSIH1IR	Reception status interrupt	CSIH1	182	$0BD0_H$	$0BD0_H$	Yes	Yes	Yes
182	ICCSIH1IC	$616C_H$	INTCSIH1IC	Communication status interrupt	CSIH1	183	$0BE0_H$	$0BE0_H$	Yes	Yes	Yes
183	ICCSIH1IJC	$616E_H$	INTCSIH1IJC	Job completion interrupt	CSIH1	184	$0BF0_H$	$0BF0_H$	Yes	Yes	Yes
184	R.F.U.	6170_H	R.F.U.			185	$0C00_H$	$0C00_H$	No	No	No
185	R.F.U.	6172_H	R.F.U.			186	$0C10_H$	$0C10_H$	No	No	No
186	R.F.U.	6174_H	R.F.U.			187	$0C20_H$	$0C20_H$	No	No	No
187	ICENCA0I0	6176_H	INTENCA0I0	Capture/compare match interrupt	ENCA0	188	$0C30_H$	$0C30_H$	No	No	Yes
188	ICENCA0I1	6178_H	INTENCA0I1	Capture/compare match interrupt	ENCA0	189	$0C40_H$	$0C40_H$	No	No	Yes
189	ICENCA0IUD	$617A_H$	INTENCA0IUD	Underflow interrupt	ENCA0	190	$0C50_H$	$0C50_H$	No	No	Yes
190	ICENCA0IEC	$617C_H$	INTENCA0IEC	Encoder clear interrupt	ENCA0	191	$0C60_H$	$0C60_H$	No	No	Yes
191	ICENCA0IOV	$617E_H$	INTENCA0IOV	Overflow interrupt	ENCA0	192	$0C70_H$	$0C70_H$	No	No	Yes
192	ICENCA1I0	6180_H	INTENCA1I0	Capture/compare match interrupt	ENCA1	193	$0C80_H$	$0C80_H$	No	No	Yes
193	ICLMA2IS	6182_H	INTLMA2IS[a]	Status interrupt	LMA2	194	$0C90_H$	$0C90_H$	Yes	Yes	Yes
194	ICLMA2IR	6184_H	INTLMA2IR[a]	Reception completion interrupt	LMA2	195	$0CA0_H$	$0CA0_H$	Yes	Yes	Yes
195	ICLMA2IT	6186_H	INTLMA2IT[a]	Transfer interrupt	LMA2	196	$0CB0_H$	$0CB0_H$	Yes	Yes	Yes
196	ICLMA3IS	6188_H	INTLMA3IS[a]	Status interrupt	LMA3	197	$0CC0_H$	$0CC0_H$	Yes	Yes	Yes
197	ICLMA3IR	$618A_H$	INTLMA3IR[a]	Reception completion interrupt	LMA3	198	$0CD0_H$	$0CD0_H$	Yes	Yes	Yes
198	ICLMA3IT	$618C_H$	INTLMA3IT[a]	Transfer interrupt	LMA3	199	$0CE0_H$	$0CE0_H$	Yes	Yes	Yes
199	R.F.U.	$618E_H$	R.F.U.			200	$0CF0_H$	$0CF0_H$	No	No	No
200	R.F.U.	6190_H	R.F.U.			201	$0D00_H$	$0D00_H$	No	No	No
201	R.F.U.	6192_H	R.F.U.			202	$0D10_H$	$0D10_H$	No	No	No
202	ICCSIH2IRE	6194_H	INTCSIH2IRE	Reception error interrupt	CSIH2	203	$0D20_H$	$0D20_H$	No	Yes	Yes

Table 5-4 EI level maskable interrupts requests (7/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address 0000...	Supported by:		
Channel	Control register: Register name	Control register: Address FFFF...	Name	Source	Unit				V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
203	ICCSIH2IR	6196_H	INTCSIH2IR	Reception status interrupt	CSIH2	204	$0D30_H$	$0D30_H$	No	Yes	Yes
204	ICCSIH2IC	6198_H	INTCSIH2IC	Communication status interrupt	CSIH2	205	$0D40_H$	$0D40_H$	No	Yes	Yes
205	ICCSIH2IJC	$619A_H$	INTCSIH2IJC	Job completion interrupt	CSIH2	206	$0D50_H$	$0D50_H$	No	Yes	Yes
206	ICMLB0CI	$619C_H$	INTMLB0CI	Channel interrupt	MLB0	207	$0D60_H$	$0D60_H$	Yes	Yes	Yes
207	ICMLB0SI	$619E_H$	INTMLB0SI	System interrupt	MLB0	208	$0D70_H$	$0D70_H$	Yes	Yes	Yes
208	ICP11	$61A0_H$	INTP11	Pin input edge detection	Port	209	$0D80_H$	$0D80_H$	No	Yes	Yes
209	ICP12	$61A2_H$	INTP12	Pin input edge detection	Port	210	$0D90_H$	$0D90_H$	No	Yes	Yes
210	ICP13	$61A4_H$	INTP13	Pin input edge detection	Port	211	$0DA0_H$	$0DA0_H$	No	Yes	Yes
211	ICP14	$61A6_H$	INTP14	Pin input edge detection	Port	212	$0DB0_H$	$0DB0_H$	No	Yes	Yes
212	ICP15	$61A8_H$	INTP15	Pin input edge detection	Port	213	$0DC0_H$	$0DC0_H$	No	Yes	Yes
213	ICETHA0SRX	$61AA_H$	INTETHA0SRX	Ethernet reception data ready interrupt	ETHA0	214	$0DD0_H$	$0DD0_H$	No	No	Yes
214	ICETHA0SCRX	$61AC_H$	INTETHA0SCRX	Ethernet packet reception interrupt	ETHA0	215	$0DE0_H$	$0DE0_H$	No	No	Yes
215	ICETHA0SCTX	$61AE_H$	INTETHA0SCTX	Ethernet packet transmission interrupt	ETHA0	216	$0DF0_H$	$0DF0_H$	No	No	Yes
216	ICETHA0RS	$61B0_H$	INTETHA0RS	Ethernet reception status interrupt	ETHA0	217	$0E00_H$	$0E00_H$	No	No	Yes
217	ICETHA0TS	$61B2_H$	INTETHA0TS	Ethernet transmission status interrupt	ETHA0	218	$0E10_H$	$0E10_H$	No	No	Yes
218	ICETHA0FS	$61B4_H$	INTETHA0FS	Ethernet FIFO status interrupt	ETHA0	219	$0E20_H$	$0E20_H$	No	No	Yes
219	ICETHA0MAC	$61B6_H$	INTETHA0MAC	Ethernet MAC interrupt	ETHA0	220	$0E30_H$	$0E30_H$	No	No	Yes
220	ICKR0	$61B8_H$	INTKR0	Key return interrupt	KR0	221	$0E40_H$	$0E40_H$	No	Yes	Yes
221	ICENCA1I1	$61BA_H$	INTENCA1I1	Capture/compare match interrupt	ENCA1	222	$0E50_H$	$0E50_H$	No	No	Yes
222	ICENCA1IUD	$61BC_H$	INTENCA1IUD	Underflow interrupt	ENCA1	223	$0E60_H$	$0E60_H$	No	No	Yes
223	ICENCA1EC	$61BE_H$	INTENCA1EC	Encoder clear interrupt	ENCA1	224	$0E70_H$	$0E70_H$	No	No	Yes
224	ICENCA1IOV	$61C0_H$	INTENCA1IOV	Overflow interrupt	ENCA1	225	$0E80_H$	$0E80_H$	No	No	Yes
225	R.F.U.	$61C2_H$	R.F.U.			226	$0E90_H$	$0E90_H$	No	No	No
226	R.F.U.	$61C4_H$	R.F.U.			227	$0EA0_H$	$0EA0_H$	No	No	No
227	R.F.U.	$61C6_H$	R.F.U.			228	$0EB0_H$	$0EB0_H$	No	No	No
228	ICDMA8	$61C8_H$	INTDMA8	DMA channel 8 transfer completion (or INTCT8[b] count match interrupt)	DMA	229	$0EC0_H$	$0EC0_H$	Yes	Yes	Yes
	ICCT8		INTCT8		DMA				Yes	Yes	Yes
229	ICDMA9	$61CA_H$	INTDMA9	DMA channel 9 transfer completion (or INTCT9[b] count match interrupt)	DMA	230	$0ED0_H$	$0ED0_H$	Yes	Yes	Yes
	ICCT9		INTCT9		DMA				Yes	Yes	Yes
230	ICDMA10	$61CC_H$	INTDMA10	DMA channel 10 transfer completion (or INTCT10[b] count match interrupt)	DMA	231	$0EE0_H$	$0EE0_H$	Yes	Yes	Yes
	ICCT10		INTCT10		DMA				Yes	Yes	Yes
231	ICDMA11	$61CE_H$	INTDMA11	DMA channel 11 transfer completion (or INTCT11[b] count match interrupt)	DMA	232	$0EF0_H$	$0EF0_H$	Yes	Yes	Yes
	ICCT11		INTCT11		DMA				Yes	Yes	Yes
232	ICDMA12	$61D0_H$	INTDMA12	DMA channel 12 transfer completion (or INTCT12[b] count match interrupt)	DMA	233	$0F00_H$	$0F00_H$	Yes	Yes	Yes
	ICCT12		INTCT12		DMA				Yes	Yes	Yes
233	ICDMA13	$61D2_H$	INTDMA13	DMA channel 13 transfer completion (or INTCT13[b] count match interrupt)	DMA	234	$0F10_H$	$0F10_H$	Yes	Yes	Yes
	ICCT13		INTCT13		DMA				Yes	Yes	Yes

Table 5-4 EI level maskable interrupts requests (8/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address 0000...	Supported by:		
Channel	Control register		Name	Source	Unit				V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
	Register name	Address FFFF...									
234	ICDMA14	$61D4_H$	INTDMA14	DMA channel 14 transfer completion (or INTCT14[b] count match interrupt)	DMA	235	$0F20_H$	$0F20_H$	Yes	Yes	Yes
	ICCT14		INTCT14		DMA				Yes	Yes	Yes
235	ICDMA15	$61D6_H$	INTDMA15	DMA channel 15 transfer completion (or INTCT15[b] count match interrupt)	DMA	236	$0F30_H$	$0F30_H$	Yes	Yes	Yes
	ICCT15		INTCT15		DMA				Yes	Yes	Yes
236	ICPM1IC	$61D8_H$	INTPM1INT	General interrupt	PCM1	237	$0F40_H$	$0F40_H$	No	Yes	Yes
237	ICPM1RXREN	$61DA_H$	INTPM1RXREN	Reception data FIFO interrupt	PCM1	238	$0F50_H$	$0F50_H$	No	Yes	Yes
238	ICPM1RXORE	$61DC_H$	INTPM1RXORE	Reception overrun error interrupt	PCM1	239	$0F60_H$	$0F60_H$	No	Yes	Yes
239	ICPM1RXURE	$61DE_H$	INTPM1RXURE	Reception underrun error interrupt	PCM1	240	$0F70_H$	$0F70_H$	No	Yes	Yes
240	ICPM1RXFRE	$61E0_H$	INTPM1RXFRE	Reception frame synchronization error interrupt	PCM1	241	$0F80_H$	$0F80_H$	No	Yes	Yes
241	ICPM1TXWEN	$61E2_H$	INTPM1TXWEN	Transmission data FIFO interrupt	PCM1	242	$0F90_H$	$0F90_H$	No	Yes	Yes
242	ICPM1TXORE	$61E4_H$	INTPM1TXORE	Transmission overrun error interrupt	PCM1	243	$0FA0_H$	$0FA0_H$	No	Yes	Yes
243	ICPM1TXURE	$61E6_H$	INTPM1TXURE	Transmission underrun error interrupt	PCM1	244	$0FB0_H$	$0FB0_H$	No	Yes	Yes
244	ICPM1TXFRE	$61E8_H$	INTPM1TXFRE	Transmission frame synchronization error interrupt	PCM1	245	$0FC0_H$	$0FC0_H$	No	Yes	Yes
245	R.F.U.	$61EA_H$	R.F.U.			246	$0FD0_H$	$0FD0_H$	No	No	No
246	R.F.U.	$61EC_H$	R.F.U.			247	$0FE0_H$	$0FE0_H$	No	No	No
247	R.F.U.	$61EE_H$	R.F.U.			248	$0FF0_H$	$0FF0_H$	No	No	No
248	ICLMA10IS	$61F0_H$	INTLMA10IS[a]	Status interrupt	LMA10	249	1000_H	1000_H	Yes	Yes	Yes
249	ICLMA10IR	$61F2_H$	INTLMA10IR[a]	Reception completion interrupt	LMA10	250	1010_H	1010_H	Yes	Yes	Yes
250	ICLMA10IT	$61F4_H$	INTLMA10IT[a]	Transfer interrupt	LMA10	251	1020_H	1020_H	Yes	Yes	Yes
251	R.F.U.	$61F6_H$	R.F.U.			252	1030_H	1030_H	No	No	No
252	R.F.U.	$61F8_H$	R.F.U.			253	1040_H	1040_H	No	No	No
253	R.F.U.	$61FA_H$	R.F.U.			254	1050_H	1050_H	No	No	No
254	R.F.U.	$61FC_H$	R.F.U.			255	1060_H	1060_H	No	No	No
255	R.F.U.	$61FE_H$	R.F.U.			256	1070_H	1070_H	No	No	No

a) INTLMAnIS, INTLMAnIR, and INTLMAnIT operate as INTURTEnTIS, INTURTEnTIR, and INTURTEnTIT, respectively.

b) Selection of the shared DMA interrupts is achieved by the DMA interrupt selection registers (DMAINTSL0 and DMAINTSL1). For details, see *5.2.3 “DMA interrupt selection” on page 305.*

5.2.2 Sharing of FE level non-maskable interrupt

FE level non-maskable interrupt (FENMI) is shared by two or more interrupt sources.

(1) WDTNMIF - WDTNMI factor register

This register shows which source caused the FE level non-maskable interrupt (FENMI).

Access This register is read-only, in 32-bit units.

Address FF45 0000_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	WDTANMI1F	WDTANMI0F	FENMIF
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 5-5 WDTNMIF register contents

Bit position	Bit name	Function
2	WDTANMI1F	Watchdog timer 1 (WDTA1TNMI) flag 0: WDTA1TNMI is not generated 1: WDTA1TNMI is generated
1	WDTANMI0F	Watchdog timer 0 (WDTA0TNMI) flag 0: WDTA0TNMI is not generated 1: WDTA0TNMI is generated
0	FENMIF	Input signal flag from NMI pin 0: FENMI is not generated 1: FENMI is generated

(2) WDTNMIFC – WDTNMI factor clear register

This register clears the FE level non-maskable interrupt flags of the WDTNMIF register.

Access This register can be read or written in 32-bit units.

Address FF45 0008$_H$

Initial value Reading this registers always returns 0000 0000$_H$.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	WDTANMI1FC	WDTANMI0FC	FENMIFC
R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W

Table 5-6 WDTNMIFC register contents

Bit position	Bit name	Function
2	WDTANMI1FC	WDTANMI1F flag clear 0: No function 1: Clear WDTNMIF.WDTANMI1F flag
1	WDTANMI0FC	WDTANMI0F flag clear 0: No function 1: Clear WDTNMIF.WDTANMI0F flag
0	FENMIFC	FENMIF flag clear 0: No function 1: Clear WDTNMIF.FENMIF flag

5.2.3 DMA interrupt selection

The DMA Controller generates two different interrupts for each DMA channel m:

- DMA channel m transfer completion interrupt (INTDMAm)
- DMA channel m count match interrupt (INTCTm)

By use of the DMA interrupt selection registers DMAINTSL0 and DMAINTSL0, one of the above interrupts can be selected as an interrupt request, and assigned separately to each DMA channel.

(1) DMAINTSL0, DMAINTSL1- DMA interrupt selection registers 0, 1

These registers are used to select the DMA interrupt to be used as an interrupt request for DMA channels 0 to 15.

Access This register can be read or written in 8-bit units.

Address FF77 0414$_H$: DMAINTSL0
FF77 0418$_H$: DMAINTSL1

Initial value 00$_H$

	7	6	5	4	3	2	1	0
DMAINTSL1	SLINTCT15	SLINTCT14	SLINTCT13	SLINTCT12	SLINTCT11	SLINTCT10	SLINTCT9	SLINTCT8
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

	7	6	5	4	3	2	1	0
DMAINTSL0	SLINTCT7	SLINTCT6	SLINTCT5	SLINTCT4	SLINTCT3	SLINTCT2	SLINTCT1	SLINTCT0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-7 DMAINTSL0, DMAINTSL1 register contents

Bit position	Bit name	Function
7 to 0	SLINTCT15to SLINTCT0	DMA channel m (m = 0 to 15) interrupt request 0: DMA channel m completion INTDMAm is used as an interrupt request. 1: DMA channel m count match INTCTm is used as an interrupt request.

5.3 Edge Detection Configuration

The external interrupts INTPm and NMI can be configured to generate an interrupt request upon a rising or falling edge or upon both edges of the external pin.

The following registers are used to specify the detection edge:

Table 5-8 External interrupt edge detection registers

Interrupt	Register
INTP0	FCLA0CTL0
INTP1	FCLA0CTL1
INTP2	FCLA0CTL2
INTP3	FCLA0CTL3
INTP4	FCLA0CTL4
INTP5	FCLA0CTL5
INTP6	FCLA0CTL6
INTP7	FCLA0CTL7
INTP8	FCLA1CTL0
INTP9	FCLA1CTL1
INTP10	FCLA1CTL2
INTP11	FCLA1CTL3
INTP12	FCLA1CTL4
INTP13	FCLA1CTL5
INTP14	FCLA1CTL6
INTP15	FCLA1CTL7
NMI	FCLA2CTL0

For details, see port filter assignment in *Chapter 2 “Pin Functions”*.

5.4 Interrupt Controller Control Registers

(1) EICn - EI level interrupt control registers (n = 0 to 255)

These registers, each of which is for a channel of EI level maskable interrupt (EIINT), are used to set control conditions for each channel. The values of bits 15 to 13, bits 11 to 8, and bits 6 to 4 must always be 0.

Update the EICn register by a byte or halfword write access.

Access This register can be read or written in 16-bit, 8-bit, or 1-bit units.

Address $FFFF6000_H$ to $FFFF61FE_H$

Initial value $008F_H$. This register is initialized by any reset.

Caution Do not access the EICn registers of the interrupt channels that are not listed in *Table 5-4 "EI level maskable interrupts requests"*.

15	14	13	12	11	10	9	8
0	0	0	EIRFn	0	0	0	0
R	R	R	R/W	R	R	R	R
7	6	5	4	3	2	1	0
EIMKn	0	0	0	EIP3n	EIP2n	EIP1n	EIP0n
R/W	R	R	R	R/W	R/W	R/W	R/W

Table 5-9 EICn register contents

Bit position	Bit name	Function
12	EIRFn	Interrupt requests flag The EIRFn bit can be written by program. Setting the EIRFn bit (1) generates an EI level maskable interrupt (EIINTn), just as when an interrupt request is acknowledged. 0: No interrupt request (initial value) 1: Interrupt request
7	EIMKn	Interrupt mask bit. When the EIMKn bit is set, the interrupt request set to the interrupt request flag (EIRFn) is masked, so that the interrupt request is not issued from that channel to the CPU core. From a channel with the EIMKn bit set, interrupt pending status is not displayed by the ICSR.PMF bit. The EIMKn bit does not mask a signal input from an interrupt input pin itself and, therefore, the corresponding interrupt request flag is set even when the EIMKn bit is set. The setting of the corresponding bit of the interrupt mask register (IMR) is also reflected. 0: Enables interrupt servicing. 1: Disables interrupt servicing (initial value).
3 to 0	EIP3n to EIP0n	These bits specify 16 levels of interrupt priorities. The highest priority is 0 and the lowest is 15. If two or more interrupt requests of EI level are generated at the same time, the interrupt source having the highest priority specified by these bits is selected and reported to the CPU core. If the priority specified by the EIP3n to EIP0n bits is the same, the source having the smallest channel number has a fixed priority and is selected.

(2) IMRm - EI level interrupt mask registers (m = 0 to 15)

These registers are a collection of the EIMKn bits of the EICn registers. Each bit of IMRm reflects the setting of the corresponding EICn.EIMKn bit. Setting IMRm reflects on the corresponding EIMKn bit.

Access This register can be read or written in 16-bit or 8-bit units.

Address

IMR0: $FFFF6400_H$	IMR1: $FFFF6402_H$
IMR2: $FFFF6404_H$	IMR3: $FFFF6406_H$
IMR4: $FFFF6408_H$	IMR5: $FFFF640A_H$
IMR6: $FFFF640C_H$	IMR7: $FFFF640E_H$
IMR8: $FFFF6410_H$	IMR9: $FFFF6412_H$
IMR10: $FFFF6414_H$	IMR11: $FFFF6416_H$
IMR12: $FFFF6418_H$	IMR13: $FFFF641A_H$
IMR14: $FFFF641C_H$	IMR15: $FFFF641E_H$

Initial value $FFFF_H$. This register is initialized by any reset.

Caution The EIMKn bits corresponding to interrupt channels that are not listed in *Table 5-4 "EI level maskable interrupts requests"* must always be set to 1.

15	14	13	12	11	10	9	8
EIMK mx16+15	EIMK mx16+14	EIMK mx16+13	EIMK mx16+12	EIMK mx16+11	EIMK mx16+10	EIMK mx16+9	EIMK mx16+8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
EIMK mx16+7	EIMK mx16+6	EIMK mx16+5	EIMK mx16+4	EIMK mx16+3	EIMK mx16+2	EIMK mx16+1	EIMK mx16+0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-10 IMRm register contents

Bit position	Bit name	Function
15 to 0	EIMK15 to EIMK0	These bits mask an interrupt from channels 0 to 15 of EI level maskable interrupt EIINT. 0: Interrupt request issued. 1: Interrupt request not issued.

(3) ISPR - In-service priority register

This register holds the interrupt priority of EI level maskable interrupt (EIINT) that is being processed by the CPU core. When a response of acknowledging an interrupt request is received from the CPU core, the bit of this register corresponding to the priority of that interrupt request is set. When it is reported by the CPU core that interrupt servicing has been completed, the bit of this register having the highest priority of those that are set is automatically cleared. The bit of this register is not cleared if execution has returned from an FE level interrupt. If multiple interrupts of EI level maskable interrupt (EIINT) are generated, the bits of this register corresponding to the priorities of the interrupts that have been acknowledged are sequentially set, and the history of the priorities of the multiple interrupts is held.

All bits of the ISPR register can be cleared at once by writing $FFFF_H$ to the ISPC register and then writing 0000_H to the ISPR register. It is not possible to specify a bit to set or clear by using software. Once a bit has been cleared, it is impossible to restore its original value.

When this register is accessed in 8-bit units, either the upper 8 bits [15:8] or the lower 8 bits [7:0] are accessed.

Access This register is read-only, in 16-bit or 8-bit units.

Address $FFFF6440_H$

Initial value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
ISPR15	ISPR14	ISPR13	ISPR12	ISPR11	ISPR10	ISPR9	ISPR8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ISPR7	ISPR6	ISPR5	ISPR4	ISPR3	ISPR2	ISPR1	ISPR0
R	R	R	R	R	R	R	R

Table 5-11 ISPR register contents

Bit position	Bit name	Function
15 to 0	ISPR15 to ISPR0	These bits indicate the priority of the interrupt being acknowledged. 0: Interrupt request of the priority corresponding to the specified bit position is not acknowledged. 1: Interrupt request of the priority corresponding to the specified bit position is being processed by the CPU core.

(4) PMR - Priority mask register

This register is used to specify an interrupt priority by which an interrupt request flag of EI level maskable interrupt (EIINT) is to be masked. It disables all at once the interrupt requests from the EIINT channel for which the interrupt priority specified by this register is set.

The position of each bit of this register corresponds to an interrupt priority. For example, if 1 is set to bit 0, channel of interrupt priority 0 can be masked.

This register can be read or written in 16-bit, 8-bit, or 1-bit units. When this register is accessed in 8-bit units, either the upper 8 bits [15:8] or the lower 8 bits [7:0] are accessed.

Access This register can be read or written in 16-bit, 8-bit, or 1-bit units.

Address $FFFF6448_H$

Initial value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
PMR15	PMR14	PMR13	PMR12	PMR11	PMR10	PMR9	PMR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
PMR7	PMR6	PMR5	PMR4	PMR3	PMR2	PMR1	PMR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-12 PMR register contents

Bit position	Bit name	Function
15 to 0	PMR15 to PMR0	These bits specify an interrupt priority by which an interrupt request flag is masked. 0: Enables interrupt servicing of the priority corresponding to the specified bit position (initial value). 1: Disables interrupt servicing of the priority corresponding to the specified bit position.

(5) ISPC - In-service priority mask register

All bits of the ISPR register can be cleared at once by writing $FFFF_H$ to the ISPC register and then writing 0000_H to the ISPR register. At the same time, all the processing modes of FE level NMI, FE level maskable interrupt (FEINT), and EI level maskable interrupt (EIINT) of the ICSR register are cancelled. As a result, the internal mode registers of INTC for interrupt servicing, which indicate that an interrupt request is being processed by the CPU core, are cleared. The contents of these registers that once have been cleared cannot be restored by software.

When the ISPR register is cleared by writing 0 to all the bits of the ISPR register, the value of the ISPC register is also automatically cleared to 0. Upon a read access to the ISPC register, after $FFFF_H$ is written to this register all bits read 1, and after a reset or after this register is cleared all bits read 0. No bit value in this register changes unless 1 or 0 is simultaneously written to all bits. If 0 is written to all the bits while they are 1, the value of all the bits of the ISPC register is cleared to 0, and the value of the ISPR register does not change.

Access This register is read-only, in 16-bit units.

Address $FFFF6450_H$

Initial value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
ISPC15	ISPC14	ISPC13	ISPC12	ISPC11	ISPC10	ISPC9	ISPC8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ISPC7	ISPC6	ISPC5	ISPC4	ISPC3	ISPC2	ISPC1	ISPC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-13 ISPC register contents

Bit position	Bit name	Function
15 to 0	ISPC15 to ISPC0	1 or 0 is read from all these bits. ISPR can be cleared if 0 is written to all the bits of ISPR when 1 is read from all the bits of this register.

(6) SCR - Selected channel hold register

This register holds the channel number of the EI level maskable interrupt (EIINT) acknowledged by the CPU. It cannot be written with software. The value of this register is updated when an interrupt vector is reported to the CPU core. Note that this register is overwritten when multiple interrupt requests of EI level INT are acknowledged. This register is not updated when an interrupt request of FE level is acknowledged.

When this register is accessed in 8-bit units, either the upper 8 bits [15:8] or the lower 8 bits [7:0] are accessed.

Access This register is read-only, in 16-bit or 8-bit units.

Address $FFFF6458_H$

Initial value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
SCR7	SCR6	SCR5	SCR4	SCR3	SCR2	SCR1	SCR0
R	R	R	R	R	R	R	R

Table 5-14 SCR register contents

Bit position	Bit name	Function
7 to 0	SCR7 to SCR0	Holds the channel number of the maskable interrupt that has been acknowledged by the CPU. The value of these bits is updated when an interrupt vector is reported to the CPU core. It is overwritten when multiple interrupts of EI level maskable interrupt (EIINT) are acknowledged. These bits are not updated when an FE level interrupt is acknowledged. Nothing happens if this register is accessed for write.

(7) ICSR: Interrupt controller status register

This register indicates the operation status of the interrupt controller. Especially, bits 2 to 0 of this register serve as a mode register of interrupt servicing. This register cannot be written with software.

When this register is accessed in 8-bit units, either the upper 8 bits [15:8] or the lower 8 bits [7:0] are accessed.

Access This register is read-only, in 16-bit or 8-bit units.

Address $FFFF645A_H$

Initial value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	PMF
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	FNR	FIR	EIR	0	FNE	FIE	EIE
R	R	R	R	R	R	R	R

Table 5-15 ICSR register contents

Bit position	Bit name	Function
8	RMF	Indicates 1 if the request flag for a channel of an EI level maskable interrupt (EIINT) that has the interrupt priority prohibited by the setting of PMR from being serviced is set.
6	FNR	Indicates 1 if an FE level non-maskable interrupt (FENMI) has been issued to the CPU.
5	FIR	Indicates 1 if an FE level maskable interrupt (FEINT) has been issued to the CPU.
4	EIR	Indicates 1 if an EI level maskable interrupt (EIINT) has been issued to the CPU.
2	FNE	Indicates 1 if an FE level non-maskable interrupt (FENMI) is being serviced by the CPU.
1	FIE	Indicates 1 if an FE level maskable interrupt (FEINT) is being serviced by the CPU.
0	EIE	Indicates 1 if an EI level maskable interrupt (EIINT) is being serviced by the CPU.

(8) FNC - FE level NMI control register

This register is used to set control conditions for the FE level non-maskable interrupt (FENMI). When this register is accessed in 8-bit units, either the upper 8 bits [15:8] or the lower 8 bits [7:0] are accessed.

Access This register is read-only, in 16-bit, 8-bit, or 1-bit units.

Address $FFFF645C_H$

Initial value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	FNRF	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

Table 5-16 FNC register contents

Bit position	Bit name	Function
12	FNRF	Interrupt request flag 0: No interrupt request (initial value) 1: Interrupt request

(9) FIC: FE level maskable interrupt control register

This register is used to set control conditions for the FE level maskable interrupt (FEINT). When this register is accessed in 8-bit units, either the upper 8 bits [15:8] or the lower 8 bits [7:0] are accessed.

Access This register is read-only, in 16-bit, 8-bit, or 1-bit units.

Address $FFFF645E_H$

Initial value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	FIRF	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

Table 5-17 FNC register contents

Bit position	Bit name	Function
12	FIRF	Interrupt request flag 0: No interrupt request (initial value) 1: Interrupt request

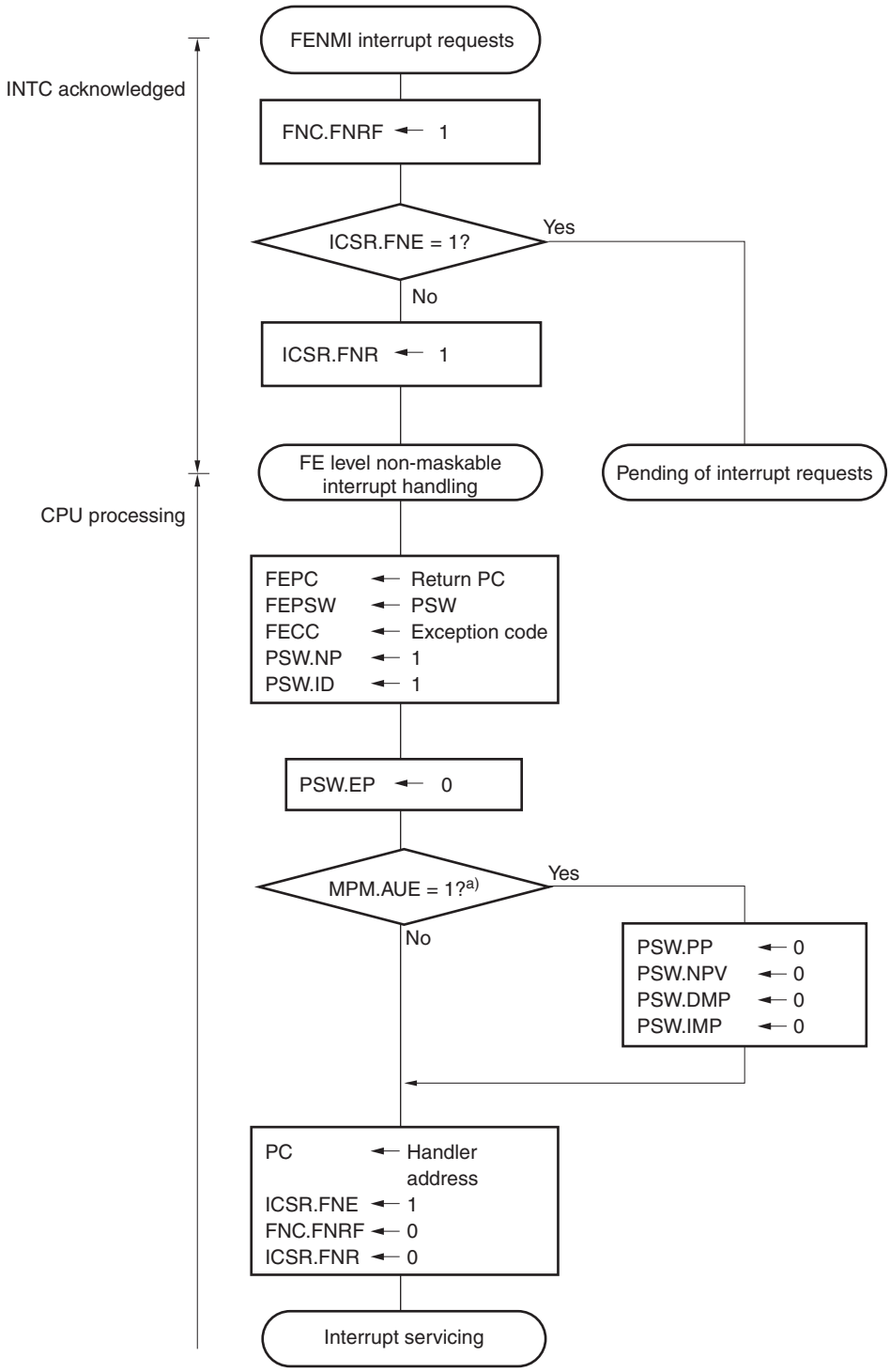
5.5 Interrupt Acknowledgment and Restoring

This section describes the operation during interrupt acknowledgment and restoring from interrupt servicing.

5.5.1 FE level non-maskable interrupt caused by FENMI interrupt request

When an FENMI interrupt is requested, an FE level non-maskable interrupt is generated in the CPU. This FE level non-maskable interrupt is used when a fatal system error occurs.

Caution Upon acknowledgment of the FENMI interrupt, generation of the next FENMI, FEINT, or EIINT interrupt is pended until the FERET instruction is executed (interrupt request is acknowledged and held.)
FENMI can be acknowledged even when the NP bit is set to 1.Therefore, if the FENMI interrupt occurs during the processing of an FEINT exception, PPI exception, or other FE level exceptions, the save address is lost and cannot be restored. After a FENMI interrupt is requested and the required processing has been completed, execute a system reset, etc. Return to the original processing is not possible.



a) If a processor protection exception (MDP or MIP) occurs, the PSW.PP, NPV, DMP, and IMP bits are always cleared (0), regardless of the status of MPM.AUE.

Figure 5-1 Processing upon occurrence of FENMI interrupt request

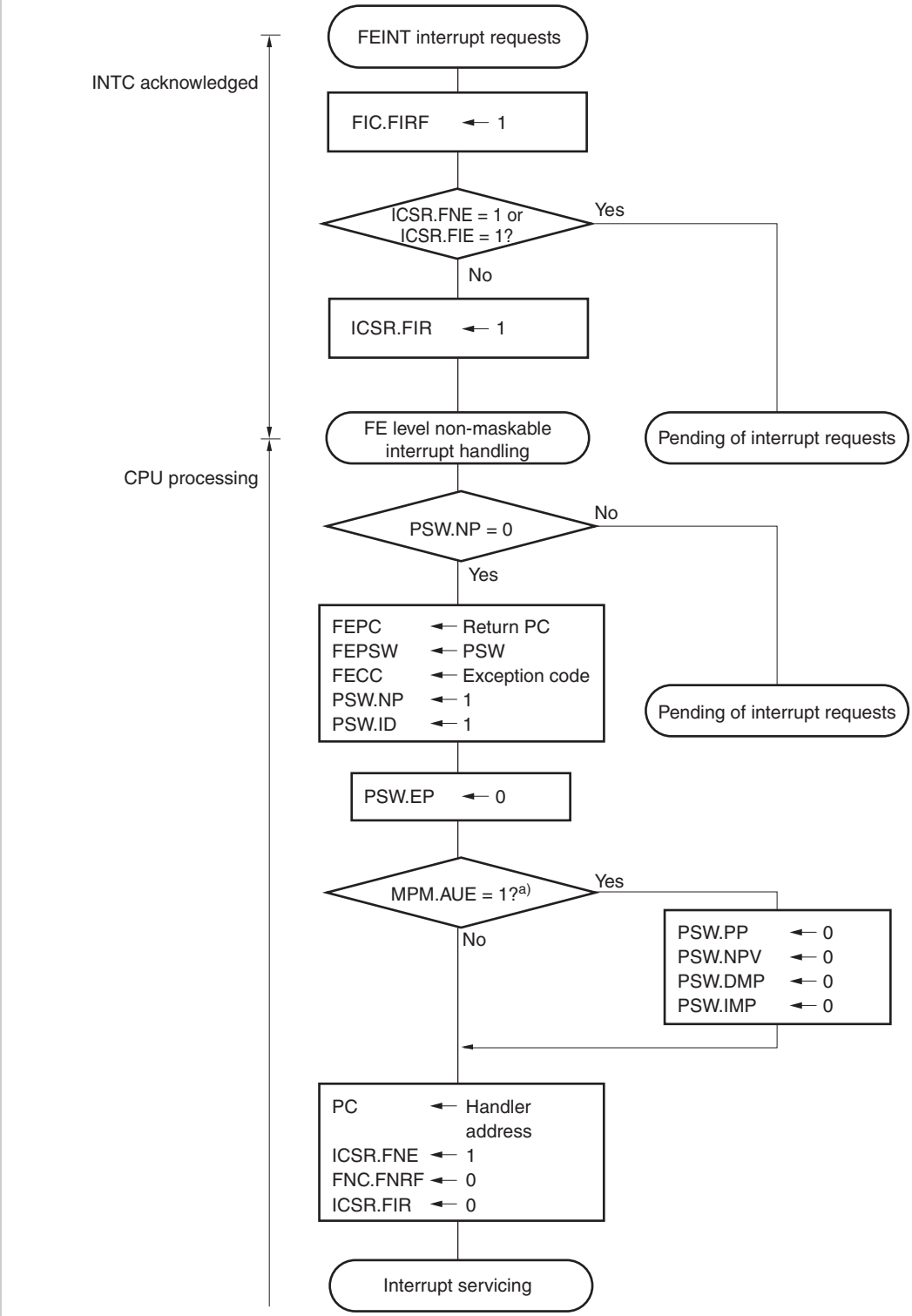
5.5.2 Restoring from FE level non-maskable interrupt (FENMI)

After the FE level non-maskable interrupt (FENMI) is serviced, it is not possible to return to the original processing because FENMI is used for fatal system errors. In this case, execute a system reset after exception processing.

5.5.3 FE level maskable interrupt caused by FEINT interrupt request

When an FEINT interrupt is requested through the FEINT pin, an FE level maskable interrupt is generated. This is an FE level interrupt after which the processor status can be recovered.

Upon acknowledgment of the FEINT interrupt, generation of the next FEINT or EIINT interrupt is pended until the FERET instruction is executed. Interrupt request is acknowledged and held.



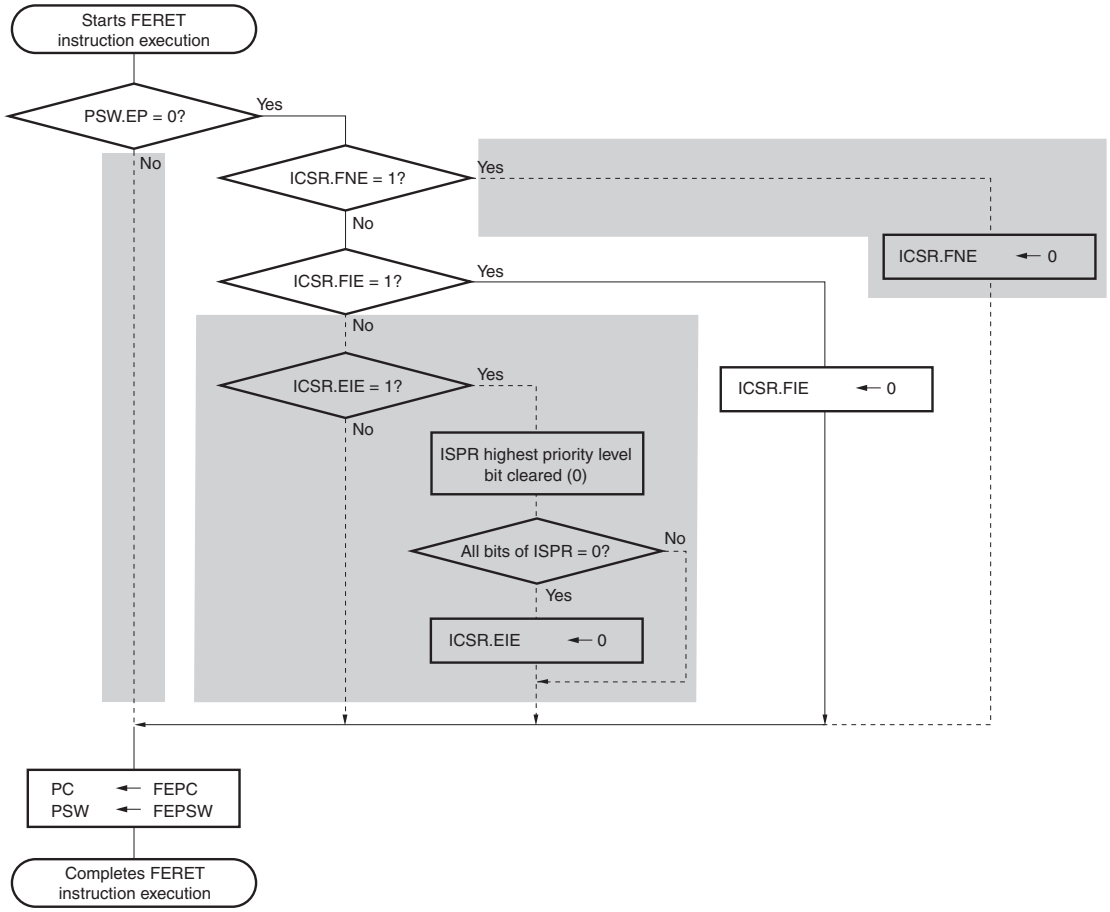
a) If a processor protection exception (MDP or MIP) occurs, the PSW.PP, NPV, DMP, and IMP bits are always cleared (0), regardless of the status of MPM.AUE.

Figure 5-2 Processing upon occurrence of FEINT interrupt request

5.5.4 Restoring from FE level maskable interrupt (FEINT) servicing

Restoring from FE level maskable interrupt (FEINT) servicing is performed using the FERET instructions. If the FERET instruction is executed while the PSW.EP bit is cleared, the restore processing from the FE level maskable interrupt (FEINT) is performed. If the FERET instruction is executed while the PSW.EP bit is 1, however, some registers such as ICSR and ISPR are not cleared, so the processing cannot be restored completely from the interrupt servicing. To return from FE level maskable interrupt (FEINT) servicing, therefore, make sure to execute the FERET instruction while the PSW.EP bit is cleared.

Caution Although RETI instructions are provided for the V850E2M CPU core to keep backward compatibility with the V850E1 and V850E2 architectures, their use is, in principle, prohibited. All RETI instructions other than those in unmodifiable existing programs must be replaced with EIRET or FERET instructions.



Note: The shaded portions indicate processing which do not branch during restoring from FE level maskable interrupt (FEINT) servicing.

Figure 5-3 Restoring from FE level maskable interrupt (FEINT) servicing

5.5.5 EI level maskable interrupt caused by EIINT interrupt request

When an EI level maskable interrupt is requested, an EIINT interrupt is requested to the CPU (the transition to the interrupt handler occurs according to the setting of the IMR register of the INTC). This is an EI level interrupt after which the processor status can be recovered.

While servicing an EIINT, the number of the channel where the interrupt was input is set to the SCR register. As a result, the channel number can be easily known when sharing the same interrupt vector among several channels.

Caution Upon acknowledgment of an EI level interrupt, the priority level of the currently acknowledged interrupt is registered to the ISPR (in-service priority) register. After this, any interrupts with a priority level lower than that of the ISPR register are not generated until the EIRET instruction is executed. Note that interrupt requests are acknowledged and held.
Registration of the priority level of the currently acknowledged interrupt and deletion of the priority level of the interrupt during EIRET to/from the ISPR register are automatically performed by the hardware. Write to the ISPR register cannot be performed by software. Write operations are ignored.

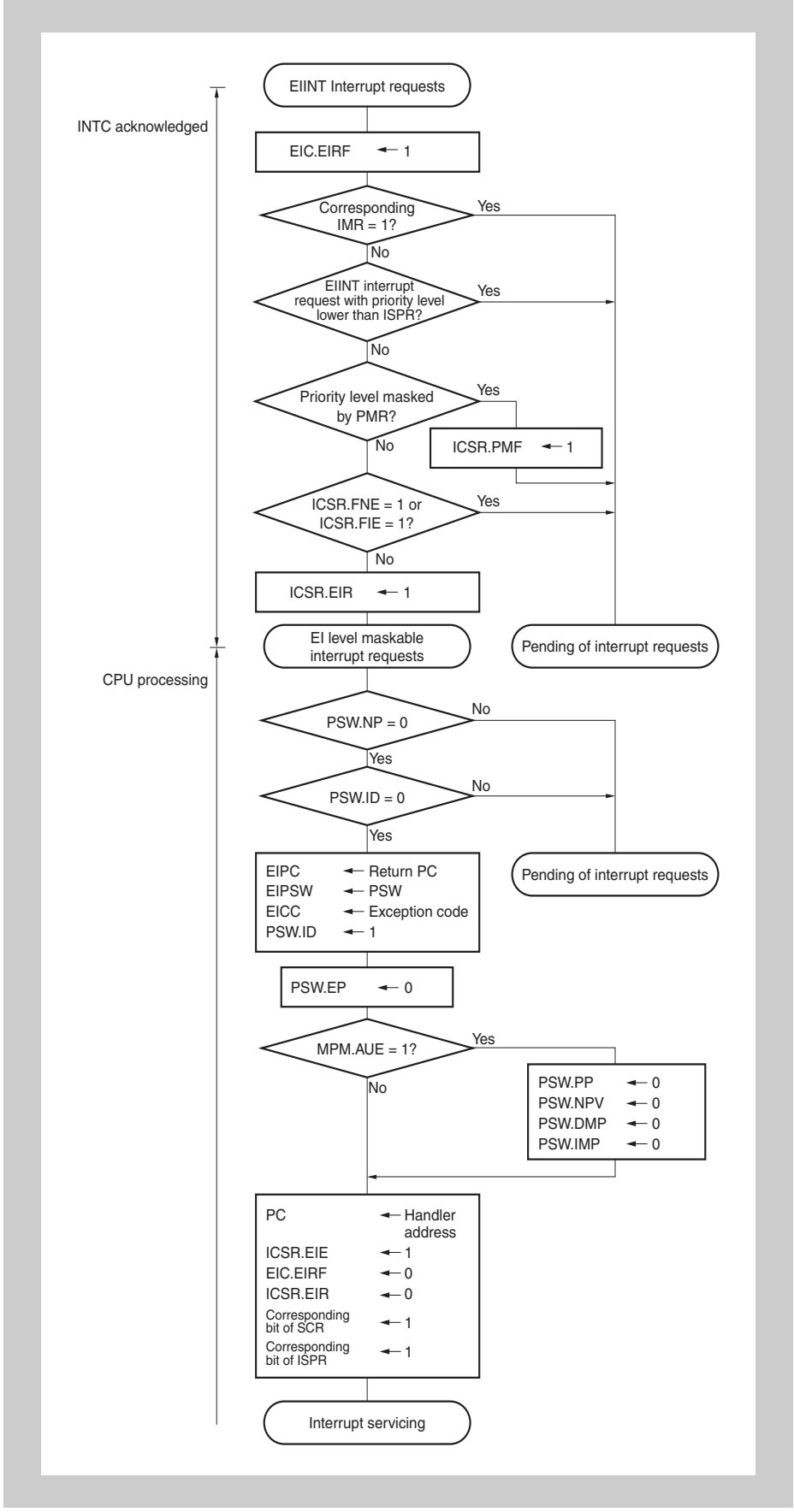
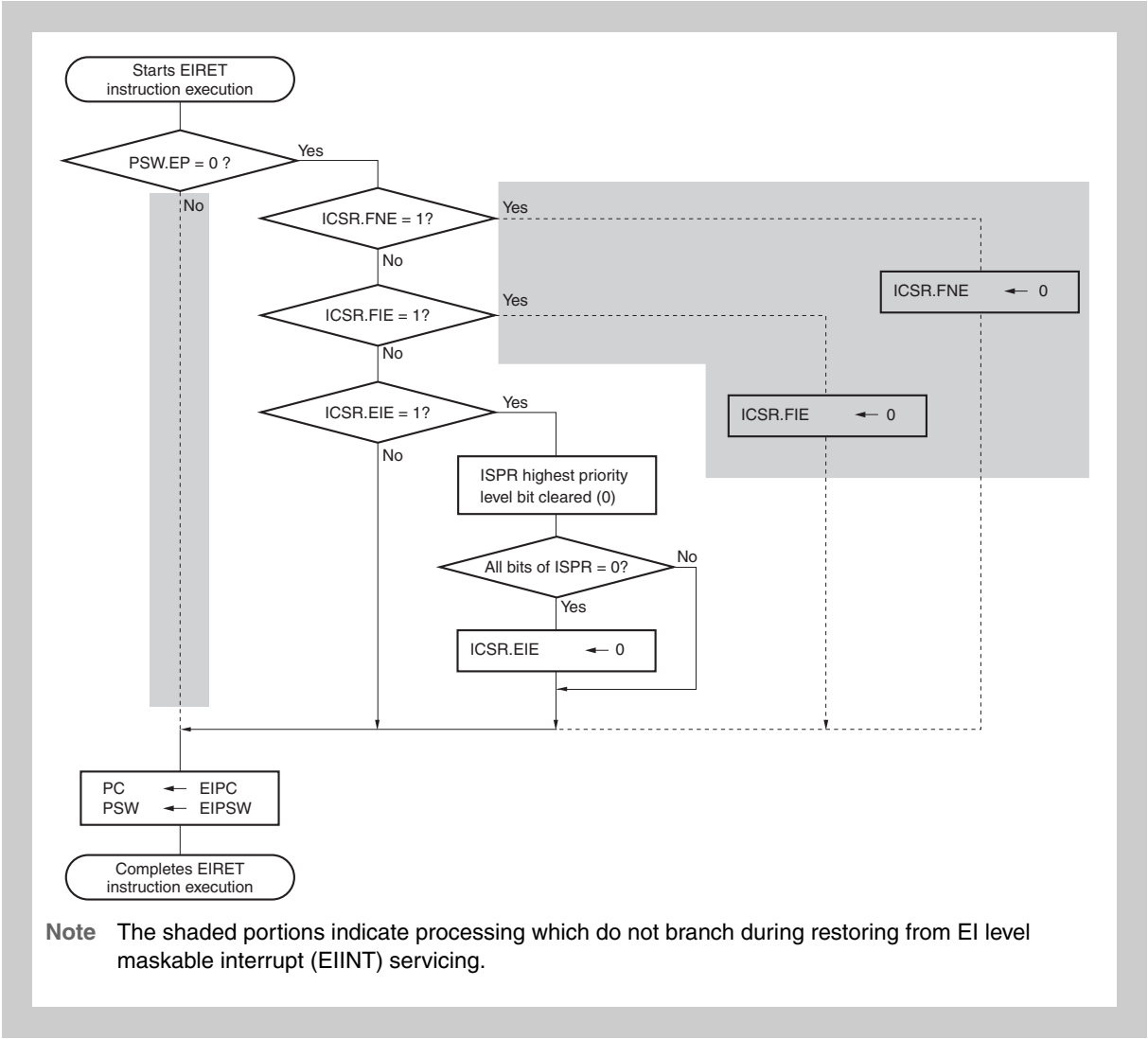


Figure 5-4 Processing upon occurrence of EIINT interrupt request

5.5.6 Restoring from EI level maskable interrupt (EIINT)

Restoring from EI level maskable interrupt (EIINT) is performed using the EIRET instruction. If the EIRET instruction is executed while the PSW.EP bit is cleared, the restore processing from the EI level maskable interrupt (EIINT) is performed. If the EIRET instruction executed while the PSW.EP bit is 1, however, some registers such as ICSR and ISPR are not cleared, so the processing cannot be restored completely from the interrupt servicing. To return from EI level maskable interrupt (EIINT) servicing, therefore, make sure to execute the EIRET instruction while the PSW.EP bit cleared.

Caution Although RETI instructions are provided for the V850E2-V3 CPU core to keep backward compatibility with the V850E1 and V850E2 architectures, their use is, in principle, prohibited. All RETI instructions other than those in the existing programs and cannot be modified must be replaced with EIRET or FERET instructions.



Note The shaded portions indicate processing which do not branch during restoring from EI level maskable interrupt (EIINT) servicing.

Figure 5-5 Restoring from EI level maskable interrupt (EIINT)

5.6 Interrupt Operation

5.6.1 Mask function of EI level maskable interrupt (EIINT)

Interrupt masking can be specified for each EINNT interrupt channel. Interrupt masking is specified by setting the following register.

EICn.EIMKn	Operation
1	Masks the interrupt
0	Enables the interrupt.

The EICn.EIMKn bits can be read and written from the corresponding EIMKn bits of the IMRm register. They share the same registers.

Operation example

1. When "1" is written to the IMRm.EIMKn bit, an interrupt to the corresponding channel is prohibited.
2. When the EICn.EIMKn bit is read, "1" is read.

Caution The EIMKn bit is used to mask the processing while an interrupt is being held. Even if the EIMKn bit is set to 1, an interrupt request is acknowledged and held. Therefore, no interrupt occurs if software requests an interrupt that is prohibited by the EIMKn bit. If the EIMKn bit is cleared while an interrupt request is held, the corresponding interrupt occurs immediately. To delete an interrupt request that is being held, clear the corresponding EIRFn bit.

5.6.2 Interrupt priority level judgment

When FENMI, FEINT, and EIINT interrupts are input, their priorities and those of the other exceptions are compared, and the exception or interrupt that has the highest priority is requested. The exceptions and interrupts requested at the same time are processed according to the pre-assigned priority order (the default priority order). The priority orders of FENMI, FEINT, and EIINT interrupts are as follows.

FENMI > FEINT > EIINT

For details about other exceptions, see *Table 5-1 “Exception cause list” on page 290* and the V850E2M Architecture User's Manual.

For EIINT interrupts, the priority level can be set independently for each interrupt source. The priority level is specified by the EIC0-EIC255.EIP3-EIP0 bits. Priority levels from 0 to 15 can be set. 0 is the highest priority level, and 15 is the lowest. If multiple EIINT interrupts that have the same priority level are requested at the same time, the interrupt whose interrupt channel number is the smallest takes priority.

Table 5-18 Example of EIINT interrupt priority level settings and priority levels

EIINT	EIP3 to EIP0 settings	Priority level during operation
EIINT0	3	10
EIINT1	4	11
EIINT2	0	1
EIINT3	0	2
EIINT4	1	3
EIINT5	2	6
EIINT6	2	7
EIINT7	1	4
EIINT8	1	5
EIINT9	2	8
EIINT10	2	9

The interrupt controller performs multiple interrupt processing in which it can acknowledge additional interrupts to that which is currently being serviced. When multiple EIINT interrupts are requested at the same time, the interrupt to be acknowledged is determined with the following procedure.

(1) Comparison with the priority level as the interrupt currently being serviced

Interrupts with the same or lower priority level as the interrupt currently being serviced are held.

The priority level of the interrupt currently being serviced is held in the ISPR register.

Interrupts with a higher priority level than the interrupt currently being serviced go on to the next priority judgment stage.

(2) Masking through priority mask register (PMR)

Only interrupts enabled by the PMR register go on to the next priority judgment stage.

(3) Of the requested interrupt sources, that with the highest priority level is selected.

If multiple interrupt sources that have the highest priority are requested at the same time, the interrupt source that has the smallest interrupt channel number is selected.

(4) Interrupt held by the CPU

Interrupt acknowledgment enters pending status according to the state of the NP and ID bits of the PSW register. At this time, priority judgment among EIINT interrupts, and priority judgment among EIINT interrupts, FEINT interrupts, and FENMI interrupts are performed even while interrupt acknowledgment is pending, and when the acknowledgment conditions are met, the interrupt that has the highest priority is selected.

Example An EIINT that has priority level 3 is requested while another EIINT that has priority level 5 has already been requested, but interrupt generation is pending because the PSW.ID bit is set. Then, if the PSW.ID bit is cleared, the EIINT with priority level 3 is generated.

Multiple interrupt processing in which another interrupt is acknowledged while an interrupt is being serviced is shown in *Figure 5-6 “Example of processing in which another interrupt request signal is issued while an interrupt is being serviced (1)” on page 326*.

When an interrupt request signal is acknowledged, the PSW.ID flag is automatically set to 1. Therefore, when multiple interrupts are to be used, clear the ID flag to 0 beforehand (for example, by placing the EI instruction in the interrupt service program) to set the interrupt enable mode.

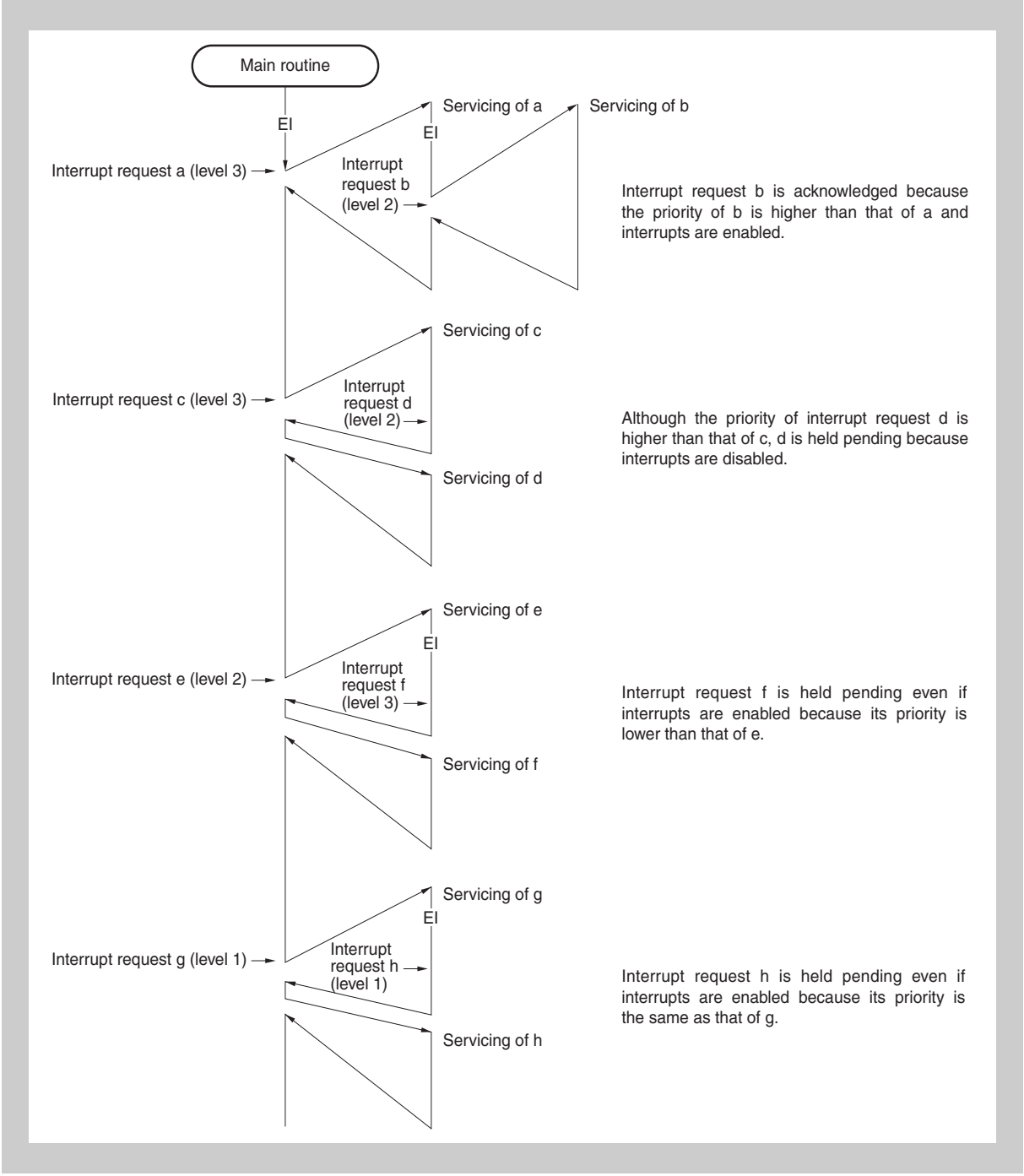


Figure 5-6 Example of processing in which another interrupt request signal is issued while an interrupt is being serviced (1)

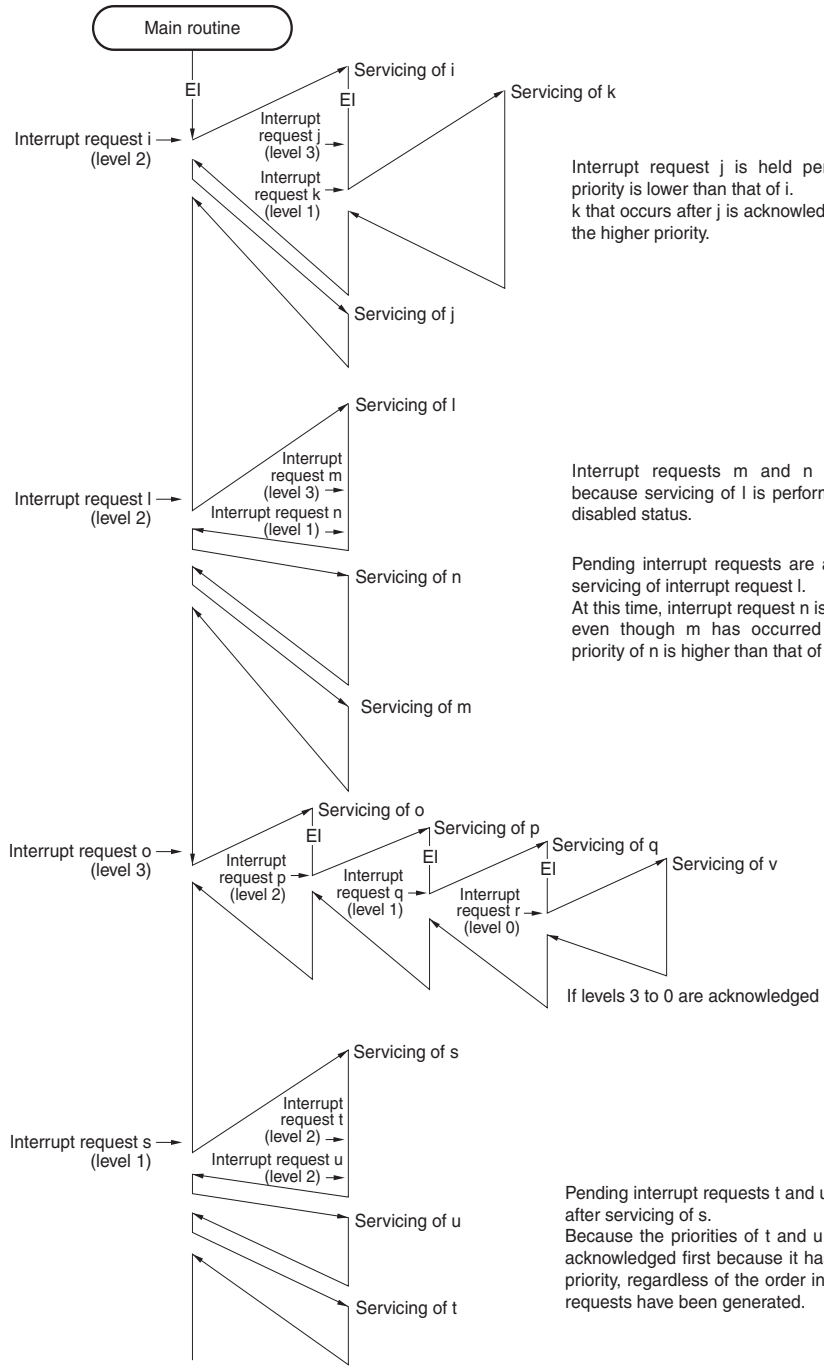


Figure 5-7 Example of processing in which another interrupt request signal is issued while an interrupt is being serviced (2)

Caution To perform multiple interrupt processing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt processing, restore the values of EIPC and EIPSW after executing the DI instruction.

Notes 1. a to u in the figure are the temporary names of interrupt request signals shown for the sake of explanation.

2. The default priority in the figure indicates the relative priority between two interrupt request signals.

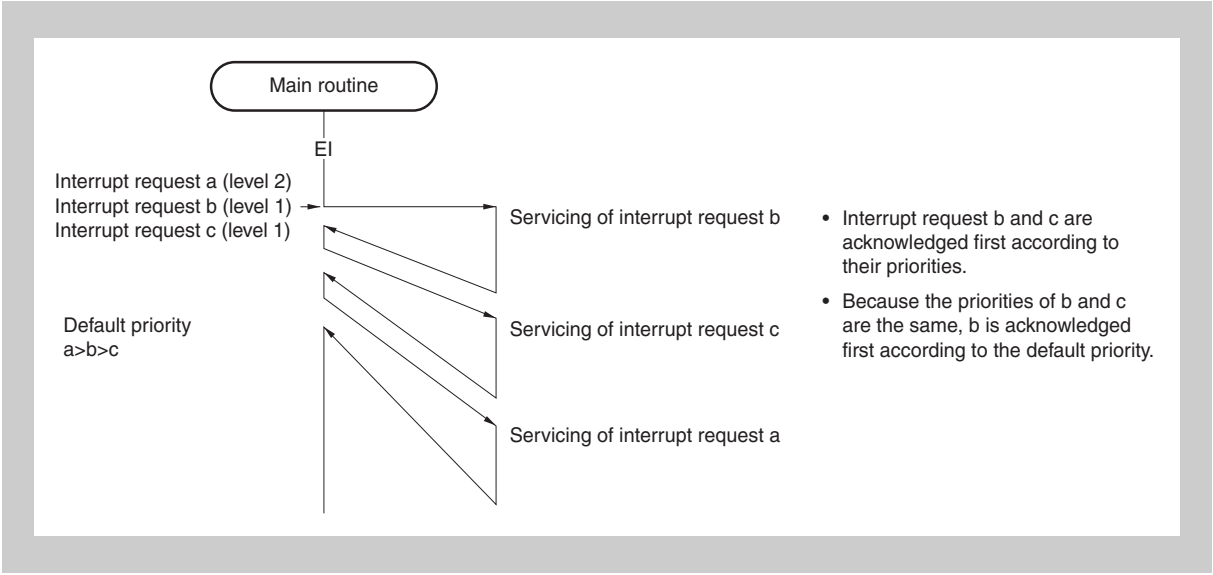


Figure 5-8 Example of servicing interrupt request signals simultaneously generated

Notes 1. a to c in the figure are the temporary names of interrupt request signals shown for the sake of explanation.

2. The default priority in the figure indicates the relative priority between two interrupt request signals.

5.6.3 Priority mask function

The priority mask function prohibits in batch EIINT interrupts of the specified interrupt priority level.

The interrupt masking priority level is specified by the PMR register. Masking and acknowledgment can be set for each priority level.

The following operations are possible using this function.

- Temporary prohibition of interrupts that have a priority level that is lower than a given priority level
- Temporary prohibition of interrupts that have a given priority level

PMR.PMRm	Operation
0	Acknowledges requests from priority level m interrupt source.
1	Masks requests from priority level m interrupt source.

Note m = 0 to 15

The PMR register prohibits interrupt occurrence. Interrupt request is acknowledged and held even while the interrupt occurrence is prohibited.

The presence of EIINT interrupts held pending with this function can be checked with *5.6.4 "Pended interrupt report function".*

5.6.4 Pended interrupt report function

The states of the currently pended interrupts can be checked with the pended interrupt report function.

This function allows checking of the following states.

- When interrupts that are masked only by the priority mask function (PMR) exist
  The ICSR.PMF bit is set to 1.
  The ICSR.PMF bit is not set to 1 only when interrupts that are priority masked through ISPR register or interrupts masked through EIMK bit exist. Thus, the existence of priority requests pended through the priority mask function can be checked while interrupts are prohibited through priority masking.

- When an EI level maskable interrupt is requested to the CPU
  The ISCR.EIR bit is set.
  Whether an EIINT interrupt request exists can be checked by checking the ICSR.EIR bit while PSW.ID = 1.

- When an FE level maskable interrupt is requested to the CPU
  The ICSR.FIR bit is set.
  Whether an FEINT interrupt request exists can be checked by checking the ICSR.FIR bit while PSW.NP = 1.

5.6.5 In-service priority clear function

This function initializes the internal status of the interrupt controller. It operates when the ISPC register is accessed. The following operations are possible using this function.

- Clear all contents of ISPR register
- Clear ICSR.EIE, FIE, and FNE bits

All bits of the ISPR register can be cleared at once by writing $FFFF_H$ to the ISPC register and then writing 0000_H to the ISPR register. Moreover, the ICSR.EIE, FIE, and FNE bits, which indicate whether the corresponding interrupt request is being processed in the CPU core, are all cleared.

The value of this register is automatically cleared by writing 0 to all bits of the ISPR register. No bit value in this register changes unless all bits are written to simultaneously.

5.7 Exception Handler Address Switching Function

Interrupt handler addresses can be switched by software.

For details, see *6.4 "Exception handler adders switching function"* in *Volume 2* of *V850E2M Architecture User's Manual*.

Chapter 6 DMA Controller (DMAC)

This chapter describes the DMA controller (DMAC).

The first section describes all V850E2/Sx4-H specific properties.

The subsequent sections describe the features that apply to all implementations.

6.1 V850E2/Sx4-H DMAC Features

Instances This microcontroller has the following number of instances of the DMAC:

Table 6-1 Instances of DMAC

DMAC	
Number of instances	16
Name	DMAC0 to DMAC15

Instances index n Throughout this chapter, the individual instances of the DMAC is identified by the index “n” (n = 0 to 15), for example, DTRSn for the DMA transfer request selection register.

(1) DMAC start sources

The DMAC start source can be selected by specifying DTFRn.IFCn[6:0].

The table below lists the DMA start sources that can be selected using the DTFRn register.

Table 6-2 DMA start sources (0 to 63)

	DTFRn.IFCn[6:0]	Interrupt to start DMA
	0	No DMA
	1	INTP0
	2	INTP1
	3	INTP2
	4	INTP3
	5	INTP4
<R>	6	INTP5
<R>	7	INTP6
<R>	8	INTP7
	9	INTP8
	10	INTP9
	11	INTP10
	12	INTTAUA0I8
	13	INTTAUA0I9
	14	INTTAUA0I10
	15	INTTAUA0I11
	16	INTTAUA0I12
	17	INTTAUA0I13
	18	INTTAUA0I14
	19	INTTAUA0I15
	20	INTTAUB2I0
	21	INTTAUB2I1
	22	INTTAUB2I4
	23	INTTAUB2I5
	24	INTTAUB2I8
	25	INTTAUB2I9
	26	INTTAUB2I12
	27	INTTAUB2I13
	28	INTIISA0ITXT
	29	INTIISA0IRXT
	30	INTIISA1ITXT
	31	INTIISA1IRXT

DTFRn.IFCn[6:0]	Interrupt to start DMA
32	INTIISA2ITXT
33	INTIISA2IRXT
34	INTIISA3ITXT
35	INTIISA3IRXT
36	INTIISA4ITXT
37	INTIISA4IRXT
38	INTADCA0I0
39	INTADCA0I1
40	INTADCA0I2
41	INTADCA0LLT
42	INTCSIG0IR
43	INTCSIG0IC
44	INTLMA0IR
45	INTLMA0IT
46	INTLMA1IR
47	INTLMA1IT
48	INTDMA0 / INTCT0[a]
49	INTDMA1 / INTCT1[a]
50	INTDMA2 / INTCT2[a]
51	INTDMA3 / INTCT3[a]
52	INTDMA4 / INTCT4[a]
53	INTDMA5 / INTCT5[a]
54	INTDMA6 / INTCT6[a]
55	INTDMA7 / INTCT7[a]
56	Setting prohibited
57	INTIICB0IA
58	INTIICB1IA
59	INTTAUJ0I0
60	INTTAUJ0I1
61	INTTAUJ0I2
62	INTTAUJ0I3
63	INTIISA5ITXT

a) The DMA interrupts are selected by using the DMAINTSL0 and DMAINTSL1 registers. For details, see *5.2.3 "DMA interrupt selection."*

Table 6-3 DMA start sources (64 to 127)

DTFRn.IFCn[6:0]	DMA trigger interrupt
64	INTIISA5IRXT
65	INTP11
66	INTP12
67	INTP13
68	INTP14
69	INTP15
70	INTTAUA0I0
71	INTTAUA0I1
72	INTTAUA0I2
73	INTTAUA0I3
74	INTIICB2IA
75	INTIICB3IA
76	INTTAUA0I4
77	INTCSIH0IC
78	INTCSIH0IJC
79	INTTAUA0I5
80	INTTAUA0I6
81	INTCSIH0IR
82	INTCSIG4IR
83	INTCSIG4IC
84	INTTAUA0I7
85	INTIEBB0D
86	INTIEBB0V
87	INTCSIH1IR
88	INTCSIH1IC
89	INTCSIH1IJC
90	Setting prohibited
91	Setting prohibited
92	INTLMA2IT
93	INTLMA10IT
94	INTLMA2IR
95	INTLMA10IR

DTFRn.IFCn[6:0]	DMA trigger interrupt
96	INTCSIH2IR
97	INTCSIH2IC
98	INTCSIH2IJC
99	INTMLB0CI
100	INTMLB0SI
101	INTETHA0SRX
102	INTETHA0SCRX
103	INTETHA0SCTX
104	INTETHA0RS
105	INTETHA0TS
106	INTETHA0FS
107	INTTAUB2I2
108	INTTAUB2I3
109	INTTAUB2I6
110	INTTAUB2I7
111	INTTAUB2I10
112	INTTAUB2I11
113	INTTAUB2I14
114	INTTAUB2I15
115	INTKR0
116	PM0TDMARQ
117	INTLMA3IR
118	INTLMA3IT
119	PM0RDMARQ
120	PM1TDMARQ
121	PM1RDMARQ
122	Setting prohibited
123	Setting prohibited
124	Setting prohibited
125	Setting prohibited
126	Setting prohibited
127	Setting prohibited

6.2 Terms

The terms used in this chapter are defined as follows.

Table 6-4 Definition of terms

Term	Function
DMA transfer	Period from the start of the first DMA cycle to assertion of INTDMA
DMA cycle	Period of transferring one unit of data (from when the lead cycle of the eLB lead cycles begins, to when the light cycle finishes. In the case of 128-bit transfer, until the read cycle is completed four times and the write cycle is completed four times)
Hardware DMA transfer request	DMA transfer request by external pins
Software DMA transfer request	DMA transfer request by internal register (DTSn.DTSnSR)
DMA transfer request	Hardware DMA transfer request or software DMA transfer request
Single transfer	The DMAC executes one DMA cycle per transfer request.
Single-step transfer	This function is only available for the DMAC. A transfer is executed the number of times specified for the transfer count setting register (DTC) per software DMA transfer request. Because the bus is released each time a transfer finishes, the CPU can generate interrupts. If a higher-priority transfer request occurs during execution of a single-step transfer, the single-step transfer is suspended while the higher-priority transfer request is executed.

6.3 Overview

Direct memory access (DMA) is used to access data without going via the CPU.

The V850E2/Sx4-H incorporates three types of DMA subsystem units: DTFR, DMAC (two units), and DMAT.DMAC. The DMAC can be used to perform high-speed data transfers.

DMAT drives the internal system bus for data access. DTFR is used to select DMA transfer factors from interrupt requests.

6.3.1 DMAC features

- Includes registers to store transfer information (transfer address and transfer size) and registers to control the DMAC.
- Outputs the acknowledged DMA transfer request to DMAT according to the contained transfer information.
- Input and outputs hardware DMA transfer requests, DMA acknowledge signals, and DMA transfer completion interrupts.
- Writes back information to registers.

6.3.2 DMA trigger factor register (DTFR) features

- The DTFR register is used to select DMA transfer factors from interrupt signals. (16 channels are selected from 128-channel interrupt signals.)

Table 6-5 Transfer sources and destinations

		Transfer destination					
		Peripheral I/O	External memory	Local RAM	HBUS-RAM	Code flash	Data flash
Transfer destination	Peripheral I/O	√	√	√	√	×	×
	External memory	√	√	√	√	×	×
	Local RAM	√	√	√	√	×	×
	HBUS-RAM	√	√	√	√	×	×
	Code flash	√	√	√	√	×	×
	Data flash	√	√	√	√	×	×

Note √: Transfer possible

×: Transfer not possible

6.3.3 DMA access memory map

For details, see *3.7 “CPU Address Map”*.

6.3.4 Prioritization of channels

The following describes how the prioritization of the DMA subsystem's various transfer channels is determined. Prioritization is determined at two stages. The first stage is when the priority level in each group among DMAC0 andDMAC1 is determined separately.

For DMAC0, the priority is specified as CH0 > CH1 > CH2 > CH3 > CH4 > CH5 > CH6 > CH7, in which CH0 has the highest priority

For DMAC1, the priority is specified as CH8 > CH9 > CH10 > CH11 > CH12 > CH13 > CH14 > CH15, in which CH8 has the highest priority.

6.3.5 Standby function

A stop acknowledge signal is returned following a stop request when the DMA transfer unit is completed.

The DMA transfer unit is one DMA cycle (from when reading starts until writing ends).

Because DMAC0 and DMAC1 each have a transfer unit, when there are two requests, a stop acknowledge signal is returned after all transfer units have been transferred.

6.4 DMAC Features

6.4.1 Features

Number of channels	8 channels x 2 groups
Transfer data size	8 bits, 16 bits, 32 bits, 128 bits
Transfer data	Fixed to little endian Misaligned data not supported
Maximum transfer count	32,768 (2^{15}) times (MSB of the 16-bit register is used for the next address function.)
Channel priority control	Fixed priority (highest priority (CH0) → lowest priority (CH15))
Subject to transfer	Cord flash, local RAM, data flash, external memory area, peripheral I/O area
Transfer type	2-cycle transfer (dual address transfer) The address at both the transfer source and destination is accessed. Two bus cycles are required to execute transfer once (read cycle + write cycle). Because the bus is not locked between the read cycle and write cycle, the CPU cycle might interrupt. When a 128-bit access is made, the write cycle is executed four times after the read cycle has been executed four times. Because the bus is not locked between the read cycles and between the write cycles, the CPU cycle might interrupt.
Transfer mode	• Single transfer mode (when a hardware DMA transfer request is generated) When a hardware DMA transfer request is generated, the bus mastership is acquired, and the bus is always released after transfer has been executed once. If another hardware DMA transfer request is generated after that, the transfer is executed. This operation is repeated until transfer has been executed the number of times specified for the transfer count register (DTC). • Single-step transfer mode (when a software DMA transfer request is generated) When a software DMA transfer request is generated, the bus mastership is acquired, and the bus is released each time a transfer has been executed. This operation is repeated until transfer has been executed the number of times specified by the transfer count register (DTC).
Transfer address control	Incremental, decremental, and fixed

Transfer error support	When data from a transfer source contains an error, or if an error occurs at the transfer destination, the DMA transfer is suspended and a SysError exception is output for the CPU.
DMA transfer request	A hardware DMA transfer request or a software DMA transfer request can be selected for each channel (by setting the DTRS register). The software DMA transfer request can be set by software (by setting the DTS register). This register also has status bits (DTS bits) that indicate that a hardware DMA transfer request has been generated. The function for notifying the INTC about peripheral I/O interrupts not assigned to DMA transfer requests depends on the system configuration. See the product's user's manual.
Transfer count match interrupt output	The transfer count compare register (DTCC) is provided for each channel and an interrupt signal (INTCT15 to INTCT0) is output upon a match with the transfer count register (DTC) on each channel.
Transfer completion interrupt output	This function outputs a transfer completion interrupt signal (INTDMA15 to INTDMA0) when a DMA transfer on each channel has been completed the number of times specified by the transfer count register (DTC).
Next address setting	A register for setting the transfer address and transfer count of the current DMA transfer and a register for setting the transfer address and transfer count of the next DMA transfer are provided for each channel. This function also has a bit for each register for specifying whether to copy the next transfer address and count to the current transfer address and count when aDMA transfer finishes.
Standby support	When a stop mode request is generated, the DMA transfer is suspended temporalily and the stop mode is entered.
DMA transfer suspension	This function supports suspending a DMA transfer by using software.

6.4.2 DMAC setting registers

Table 6-6 DMAC setting registers

(1/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF7300H	DTRC0	DMA transfer request control register 0	R/W	√	√			00H
FFFF7310H	DTRS0	DMA transfer request selection register CH0				√		0000H
FFFF7314H	DSA0	DMA source address register CH0					√	00000000H
FFFF7314H	DSA0L	DMA source address register LCH0				√		0000H
FFFF7316H	DSA0H	DMA source address register HCH0				√		0000H
FFFF7318H	DSC0	DMA source chip select register CH0				√		0001H
FFFF731CH	DNSA0	DMA next source address register CH0					√	00000000H
FFFF731CH	DNSA0L	DMA next source address register LCH0				√		0000H
FFFF731EH	DNSA0H	DMA next source address register HCH0				√		0000H
FFFF7320H	DNSC0	DMA next source chip select register CH0				√		0001H
FFFF7324H	DDA0	DMA destination address register CH0					√	00000000H
FFFF7324H	DDA0L	DMA destination address register LCH0				√		0000H
FFFF7326H	DDA0H	DMA destination address register HCH0				√		0000H
FFFF7328H	DDC0	DMA destination chip select register CH0				√		0001H
FFFF732CH	DNDA0	DMA next destination address register CH0					√	00000000H
FFFF732CH	DNDA0L	DMA next destination address register LCH0				√		0000H
FFFF732EH	DNDA0H	DMA next destination address register HCH0				√		0000H
FFFF7330H	DNDC0	DMA next destination chip select register CH0				√		0001H
FFFF7332H	DTC0	DMA transfer count register CH0				√		0000H
FFFF7334H	DNTC0	DMA next transfer count register CH0				√		0000H
FFFF7336H	DTCC0	DMA transfer count compare register CH0				√		0000H
FFFF7338H	DTCT0	DMA transfer control register CH0				√		0000H
FFFF733AH	DTS0	DMA transfer status register CH0		√	√			00H
FFFF7340H	DTRS1	DMA transfer request selection register CH1				√		0000H
FFFF7344H	DSA1	DMA source address register CH1					√	00000000H
FFFF7344H	DSA1L	DMA source address register LCH1				√		0000H
FFFF7346H	DSA1H	DMA source address register HCH1				√		0000H
FFFF7348H	DSC1	DMA source chip select register CH1				√		0001H
FFFF734CH	DNSA1	DMA next source address register CH1					√	00000000H
FFFF734CH	DNSA1L	DMA next source address register LCH1				√		0000H
FFFF734EH	DNSA1H	DMA next source address register HCH1				√		0000H

(2/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF7350H	DNSC1	DMA next source chip select register CH1	R/W			√		0001H
FFFF7354H	DDA1	DMA destination address register CH1					√	00000000H
FFFF7354H	DDA1L	DMA destination address register LCH1				√		0000H
FFFF7356H	DDA1H	DMA destination address register HCH1				√		0000H
FFFF7358H	DDC1	DMA destination chip select register CH1				√		0001H
FFFF735CH	DNDA1	DMA next destination address register CH1					√	00000000H
FFFF735CH	DNDA1L	DMA next destination address register LCH1				√		0000H
FFFF735EH	DNDA1H	DMA next destination address register HCH1				√		0000H
FFFF7360H	DNDC1	DMA next destination chip select register CH1				√		0001H
FFFF7362H	DTC1	DMA transfer count register CH1				√		0000H
FFFF7364H	DNTC1	DMA next transfer count register CH1				√		0000H
FFFF7366H	DTCC1	DMA transfer count compare register CH1				√		0000H
FFFF7368H	DTCT1	DMA transfer control register CH1				√		0000H
FFFF736AH	DTS1	DMA transfer status register CH1		√	√			00H
FFFF7370H	DTRS2	DMA transfer request selection register CH2				√		0000H
FFFF7374H	DSA2	DMA source address register CH2					√	00000000H
FFFF7374H	DSA2L	DMA source address register LCH2				√		0000H
FFFF7376H	DSA2H	DMA source address register HCH2				√		0000H
FFFF7378H	DSC2	DMA source chip select register CH2				√		0001H
FFFF737CH	DNSA2	DMA next source address register CH2					√	00000000H
FFFF737CH	DNSA2L	DMA next source address register LCH2				√		0000H
FFFF737EH	DNSA2H	DMA next source address register HCH2				√		0000H
FFFF7380H	DNSC2	DMA next source chip select register CH2				√		0001H
FFFF7384H	DDA2	DMA destination address register CH2					√	00000000H
FFFF7384H	DDA2L	DMA destination address register LCH2				√		0000H
FFFF7386H	DDA2H	DMA destination address register HCH2				√		0000H
FFFF7388H	DDC2	DMA destination chip select register CH2				√		0001H
FFFF738CH	DNDA2	DMA next destination address register CH2					√	00000000H
FFFF738CH	DNDA2L	DMA next destination address register LCH2				√		0000H
FFFF738EH	DNDA2H	DMA next destination address register HCH2				√		0000H
FFFF7390H	DNDC2	DMA next destination chip select register CH2				√		0001H
FFFF7392H	DTC2	DMA transfer count register CH2				√		0000H

(3/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF7394H	DNTC2	DMA next transfer count register CH2	R/W			√		0000H
FFFF7396H	DTCC2	DMA transfer count compare register CH2				√		0000H
FFFF7398H	DTCT2	DMA transfer control register CH2				√		0000H
FFFF739AH	DTS2	DMA transfer status register CH2		√	√			00H
FFFF73A0H	DTRS3	DMA transfer request selection register CH3				√		0000H
FFFF73A4H	DSA3	DMA source address register CH3					√	00000000H
FFFF73A4H	DSA3L	DMA source address register LCH3				√		0000H
FFFF73A6H	DSA3H	DMA source address register HCH3				√		0000H
FFFF73A8H	DSC3	DMA source chip select register CH3				√		0001H
FFFF73ACH	DNSA3	DMA next source address register CH3					√	00000000H
FFFF73ACH	DNSA3L	DMA next source address register LCH3				√		0000H
FFFF73AEH	DNSA3H	DMA next source address register HCH3				√		0000H
FFFF73B0H	DNSC3	DMA next source chip select register CH3				√		0001H
FFFF73B4H	DDA3	DMA destination address register CH3					√	00000000H
FFFF73B4H	DDA3L	DMA destination address register LCH3				√		0000H
FFFF73B6H	DDA3H	DMA destination address register HCH3				√		0000H
FFFF73B8H	DDC3	DMA destination chip select register CH3				√		0001H
FFFF73BCH	DNDA3	DMA next destination address register CH3					√	00000000H
FFFF73BCH	DNDA3L	DMA next destination address register LCH3				√		0000H
FFFF73BEH	DNDA3H	DMA next destination address register HCH3				√		0000H
FFFF73C0H	DNDC3	DMA next destination chip select register CH3				√		0001H
FFFF73C2H	DTC3	DMA transfer count register CH3				√		0000H
FFFF73C4H	DNTC3	DMA next transfer count register CH3				√		0000H
FFFF73C6H	DTCC3	DMA transfer count compare register CH3				√		0000H
FFFF73C8H	DTCT3	DMA transfer control register CH3				√		0000H
FFFF73CAH	DTS3	DMA transfer status register CH3		√	√			00H
FFFF73D0H	DTRS4	DMA transfer request selection register CH4				√		0000H
FFFF73D4H	DSA4	DMA source address register CH4					√	00000000H
FFFF73D4H	DSA4L	DMA source address register LCH4				√		0000H
FFFF73D6H	DSA4H	DMA source address register HCH4				√		0000H
FFFF73D8H	DSC4	DMA source chip select register CH4				√		0001H
FFFF73DCH	DNSA4	DMA next source address register CH4					√	00000000H
FFFF73DCH	DNSA4L	DMA next source address register LCH4				√		0000H
FFFF73DEH	DNSA4H	DMA next source address register HCH4				√		0000H
FFFF73E0H	DNSC4	DMA next source chip select register CH4				√		0001H

(4/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF73E4H	DDA4	DMA destination address register CH4	R/W				√	00000000H
FFFF73E4H	DDA4L	DMA destination address register LCH4				√		0000H
FFFF73E6H	DDA4H	DMA destination address register HCH4				√		0000H
FFFF73E8H	DDC4	DMA destination chip select register CH4				√		0001H
FFFF73ECH	DNDA4	DMA next destination address register CH4					√	00000000H
FFFF73ECH	DNDA4L	DMA next destination address register LCH4				√		0000H
FFFF73EEH	DNDA4H	DMA next destination address register HCH4				√		0000H
FFFF73F0H	DNDC4	DMA next destination chip select register CH4				√		0001H
FFFF73F2H	DTC4	DMA transfer count register CH4				√		0000H
FFFF73F4H	DNTC4	DMA next transfer count register CH4				√		0000H
FFFF73F6H	DTCC4	DMA transfer count compare register CH4				√		0000H
FFFF73F8H	DTCT4	DMA transfer control register CH4				√		0000H
FFFF73FAH	DTS4	DMA transfer status register CH4		√	√			00H
FFFF7400H	DTRS5	DMA transfer request selection register CH5				√		0000H
FFFF7404H	DSA5	DMA source address register CH5					√	00000000H
FFFF7404H	DSA5L	DMA source address register LCH5				√		0000H
FFFF7406H	DSA5H	DMA source address register HCH5				√		0000H
FFFF7408H	DSC5	DMA source chip select register CH5				√		0001H
FFFF740CH	DNSA5	DMA next source address register CH5					√	00000000H
FFFF740CH	DNSA5L	DMA next source address register LCH5				√		0000H
FFFF740EH	DNSA5H	DMA next source address register HCH5				√		0000H
FFFF7410H	DNSC5	DMA next source chip select register CH5				√		0001H
FFFF7414H	DDA5	DMA destination address register CH5					√	00000000H
FFFF7414H	DDA5L	DMA destination address register LCH5				√		0000H
FFFF7416H	DDA5H	DMA destination address register HCH5				√		0000H
FFFF7418H	DDC5	DMA destination chip select register CH5				√		0001H
FFFF741CH	DNDA5	DMA next destination address register CH5					√	00000000H
FFFF741CH	DNDA5L	DMA next destination address register LCH5				√		0000H
FFFF741EH	DNDA5H	DMA next destination address register HCH5				√		0000H
FFFF7420H	DNDC5	DMA next destination chip select register CH5				√		0001H
FFFF7422H	DTC5	DMA transfer count register CH5				√		0000H
FFFF7424H	DNTC5	DMA next transfer count register CH5				√		0000H
FFFF7426H	DTCC5	DMA transfer count compare register CH5				√		0000H

(5/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF7428H	DTCT5	DMA transfer control register CH5	R/W			√		0000H
FFFF742AH	DTS5	DMA transfer status register CH5		√	√			00H
FFFF7430H	DTRS6	DMA transfer request selection register CH6				√		0000H
FFFF7434H	DSA6	DMA source address register CH6					√	00000000H
FFFF7434H	DSA6L	DMA source address register LCH6				√		0000H
FFFF7436H	DSA6H	DMA source address register HCH6				√		0000H
FFFF7438H	DSC6	DMA source chip select register CH6				√		0001H
FFFF743CH	DNSA6	DMA next source address register CH6					√	00000000H
FFFF743CH	DNSA6L	DMA next source address register LCH6				√		0000H
FFFF743EH	DNSA6H	DMA next source address register HCH6				√		0000H
FFFF7440H	DNSC6	DMA next source chip select register CH6				√		0001H
FFFF7444H	DDA6	DMA destination address register CH6					√	00000000H
FFFF7444H	DDA6L	DMA destination address register LCH6				√		0000H
FFFF7446H	DDA6H	DMA destination address register HCH6				√		0000H
FFFF7448H	DDC6	DMA destination chip select register CH6				√		0001H
FFFF744CH	DNDA6	DMA next destination address register CH6					√	00000000H
FFFF744CH	DNDA6L	DMA next destination address register LCH6				√		0000H
FFFF744EH	DNDA6H	DMA next destination address register HCH6				√		0000H
FFFF7450H	DNDC6	DMA next destination chip select register CH6				√		0001H
FFFF7452H	DTC6	DMA transfer count register CH6				√		0000H
FFFF7454H	DNTC6	DMA next transfer count register CH6				√		0000H
FFFF7456H	DTCC6	DMA transfer count compare register CH6				√		0000H
FFFF7458H	DTCT6	DMA transfer control register CH6				√		0000H
FFFF745AH	DTS6	DMA transfer status register CH6		√	√			00H
FFFF7460H	DTRS7	DMA transfer request selection register CH7				√		0000H
FFFF7464H	DSA7	DMA source address register CH7					√	00000000H
FFFF7464H	DSA7L	DMA source address register LCH7				√		0000H
FFFF7466H	DSA7H	DMA source address register HCH7				√		0000H
FFFF7468H	DSC7	DMA source chip select register CH7				√		0001H
FFFF746CH	DNSA7	DMA next source address register CH7					√	00000000H
FFFF746CH	DNSA7L	DMA next source address register LCH7				√		0000H
FFFF746EH	DNSA7H	DMA next source address register HCH7				√		0000H
FFFF7470H	DNSC7	DMA next source chip select register CH7				√		0001H
FFFF7474H	DDA7	DMA destination address register CH7					√	00000000H
FFFF7474H	DDA7L	DMA destination address register LCH7				√		0000H
FFFF7476H	DDA7H	DMA destination address register HCH7				√		0000H

(6/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF7478H	DDC7	DMA destination chip select register CH7	R/W			√		0001H
FFFF747CH	DNDA7	DMA next destination address register CH7					√	00000000H
FFFF747CH	DNDA7L	DMA next destination address register LCH7				√		0000H
FFFF747EH	DNDA7H	DMA next destination address register HCH7				√		0000H
FFFF7480H	DNDC7	DMA next destination chip select register CH7				√		0001H
FFFF7482H	DTC7	DMA transfer count register CH7				√		0000H
FFFF7484H	DNTC7	DMA next transfer count register CH7				√		0000H
FFFF7486H	DTCC7	DMA transfer count compare register CH7				√		0000H
FFFF7488H	DTCT7	DMA transfer control register CH7				√		0000H
FFFF748AH	DTS7	DMA transfer status register CH7		√	√			00H
FFFF7500H	DTRC1	DMA transfer request control register 1		√	√			00H
FFFF7510H	DTRS8	DMA transfer request selection register CH8				√		0000H
FFFF7514H	DSA8	DMA source address register CH8					√	00000000H
FFFF7514H	DSA8L	DMA source address register LCH8				√		0000H
FFFF7516H	DSA8H	DMA source address register HCH8				√		0000H
FFFF7518H	DSC8	DMA source chip select register CH8				√		0001H
FFFF751CH	DNSA8	DMA next source address register CH8					√	00000000H
FFFF751CH	DNSA8L	DMA next source address register LCH8				√		0000H
FFFF751EH	DNSA8H	DMA next source address register HCH8				√		0000H
FFFF7520H	DNSC8	DMA next source chip select register CH8				√		0001H
FFFF7524H	DDA8	DMA destination address register CH8					√	00000000H
FFFF7524H	DDA8L	DMA destination address register LCH8				√		0000H
FFFF7526H	DDA8H	DMA destination address register HCH8				√		0000H
FFFF7528H	DDC8	DMA destination chip select register CH8				√		0001H
FFFF752CH	DNDA8	DMA next destination address register CH8					√	00000000H
FFFF752CH	DNDA8L	DMA next destination address register LCH8				√		0000H
FFFF752EH	DNDA8H	DMA next destination address register HCH8				√		0000H
FFFF7530H	DNDC8	DMA next destination chip select register CH8				√		0001H
FFFF7532H	DTC8	DMA transfer count register CH8				√		0000H
FFFF7534H	DNTC8	DMA next transfer count register CH8				√		0000H
FFFF7536H	DTCC8	DMA transfer count compare register CH8				√		0000H
FFFF7538H	DTCT8	DMA transfer control register CH8				√		0000H
FFFF753AH	DTS8	DMA transfer status register CH8		√	√			00H

(7/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF7540H	DTRS9	DMA transfer request selection register CH9	R/W			√		0000H
FFFF7544H	DSA9	DMA source address register CH9					√	00000000H
FFFF7544H	DSA9L	DMA source address register LCH9				√		0000H
FFFF7546H	DSA9H	DMA source address register HCH9				√		0000H
FFFF7548H	DSC9	DMA source chip select register CH9				√		0001H
FFFF754CH	DNSA9	DMA next source address register CH9					√	00000000H
FFFF754CH	DNSA9L	DMA next source address register LCH9				√		0000H
FFFF754EH	DNSA9H	DMA next source address register HCH9				√		0000H
FFFF7550H	DNSC9	DMA next source chip select register CH9				√		0001H
FFFF7554H	DDA9	DMA destination address register CH9					√	00000000H
FFFF7554H	DDA9L	DMA destination address register LCH9				√		0000H
FFFF7556H	DDA9H	DMA destination address register HCH9				√		0000H
FFFF7558H	DDC9	DMA destination chip select register CH9				√		0001H
FFFF755CH	DNDA9	DMA next destination address register CH9					√	00000000H
FFFF755CH	DNDA9L	DMA next destination address register LCH9				√		0000H
FFFF755EH	DNDA9H	DMA next destination address register HCH9				√		0000H
FFFF7560H	DNDC9	DMA next destination chip select register CH9				√		0001H
FFFF7562H	DTC9	DMA transfer count register CH9				√		0000H
FFFF7564H	DNTC9	DMA next transfer count register CH9				√		0000H
FFFF7566H	DTCC9	DMA transfer count compare register CH9				√		0000H
FFFF7568H	DTCT9	DMA transfer control register CH9				√		0000H
FFFF756AH	DTS9	DMA transfer status register CH9		√	√			00H
FFFF7570H	DTRS10	DMA transfer request selection register CH10				√		0000H
FFFF7574H	DSA10	DMA source address register CH10					√	00000000H
FFFF7574H	DSA10L	DMA source address register LCH10				√		0000H
FFFF7576H	DSA10H	DMA source address register HCH10				√		0000H
FFFF7578H	DSC10	DMA source chip select register CH10				√		0001H
FFFF757CH	DNSA10	DMA next source address register CH10					√	00000000H
FFFF757CH	DNSA10L	DMA next source address register LCH10				√		0000H
FFFF757EH	DNSA10H	DMA next source address register HCH10				√		0000H
FFFF7580H	DNSC10	DMA next source chip select register CH10				√		0001H
FFFF7584H	DDA10	DMA destination address register CH10					√	00000000H
FFFF7584H	DDA10L	DMA destination address register LCH10				√		0000H
FFFF7586H	DDA10H	DMA destination address register HCH10				√		0000H
FFFF7588H	DDC10	DMA destination chip select register CH10				√		0001H

(8/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF758CH	DNDA10	DMA next destination address register CH10	R/W				√	00000000H
FFFF758CH	DNDA10L	DMA next destination address register LCH10				√		0000H
FFFF758EH	DNDA10H	DMA next destination address register HCH10				√		0000H
FFFF7590H	DNDC10	DMA next destination chip select register CH10				√		0001H
FFFF7592H	DTC10	DMA transfer count register CH10				√		0000H
FFFF7594H	DNTC10	DMA next transfer count register CH10				√		0000H
FFFF7596H	DTCC10	DMA transfer count compare register CH10				√		0000H
FFFF7598H	DTCT10	DMA transfer control register CH10				√		0000H
FFFF759AH	DTS10	DMA transfer status register CH10		√	√			00H
FFFF75A0H	DTRS11	DMA transfer request selection register CH11				√		0000H
FFFF75A4H	DSA11	DMA source address register CH11					√	00000000H
FFFF75A4H	DSA11L	DMA source address register LCH11				√		0000H
FFFF75A6H	DSA11H	DMA source address register HCH11				√		0000H
FFFF75A8H	DSC11	DMA source chip select register CH11				√		0001H
FFFF75ACH	DNSA11	DMA next source address register CH11					√	00000000H
FFFF75ACH	DNSA11L	DMA next source address register LCH11				√		0000H
FFFF75AEH	DNSA11H	DMA next source address register HCH11				√		0000H
FFFF75B0H	DNSC11	DMA next source chip select register CH11				√		0001H
FFFF75B4H	DDA11	DMA destination address register CH11					√	00000000H
FFFF75B4H	DDA11L	DMA destination address register LCH11				√		0000H
FFFF75B6H	DDA11H	DMA destination address register HCH11				√		0000H
FFFF75B8H	DDC11	DMA destination chip select register CH11				√		0001H
FFFF75BCH	DNDA11	DMA next destination address register CH11					√	00000000H
FFFF75BCH	DNDA11L	DMA next destination address register LCH11				√		0000H
FFFF75BEH	DNDA11H	DMA next destination address register HCH11				√		0000H
FFFF75C0H	DNDC11	DMA next destination chip select register CH11				√		0001H
FFFF75C2H	DTC11	DMA transfer count register CH11				√		0000H
FFFF75C4H	DNTC11	DMA next transfer count register CH11				√		0000H
FFFF75C6H	DTCC11	DMA transfer count compare register CH11				√		0000H
FFFF75C8H	DTCT11	DMA transfer control register CH11				√		0000H
FFFF75CAH	DTS11	DMA transfer status register CH11		√	√			00H
FFFF75D0H	DTRS12	DMA transfer request selection register CH12				√		0000H

(9/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF75D4H	DSA12	DMA source address register CH12	R/W				√	00000000H
FFFF75D4H	DSA12L	DMA source address register LCH12				√		0000H
FFFF75D6H	DSA12H	DMA source address register HCH12				√		0000H
FFFF75D8H	DSC12	DMA source chip select register CH12				√		0001H
FFFF75DCH	DNSA12	DMA next source address register CH12					√	00000000H
FFFF75DCH	DNSA12L	DMA next source address register LCH12				√		0000H
FFFF75DEH	DNSA12H	DMA next source address register HCH12				√		0000H
FFFF75E0H	DNSC12	DMA next source chip select register CH12				√		0001H
FFFF75E4H	DDA12	DMA destination address register CH12					√	00000000H
FFFF75E4H	DDA12L	DMA destination address register LCH12				√		0000H
FFFF75E6H	DDA12H	DMA destination address register HCH12				√		0000H
FFFF75E8H	DDC12	DMA destination chip select register CH12				√		0001H
FFFF75ECH	DNDA12	DMA next destination address register CH12					√	00000000H
FFFF75ECH	DNDA12L	DMA next destination address register LCH12				√		0000H
FFFF75EEH	DNDA12H	DMA next destination address register HCH12				√		0000H
FFFF75F0H	DNDC12	DMA next destination chip select register CH12				√		0001H
FFFF75F2H	DTC12	DMA transfer count register CH12				√		0000H
FFFF75F4H	DNTC12	DMA next transfer count register CH12				√		0000H
FFFF75F6H	DTCC12	DMA transfer count compare register CH12				√		0000H
FFFF75F8H	DTCT12	DMA transfer control register CH12				√		0000H
FFFF75FAH	DTS12	DMA transfer status register CH12		√	√			00H
FFFF7600H	DTRS13	DMA transfer request selection register CH13				√		0000H
FFFF7604H	DSA13	DMA source address register CH13					√	00000000H
FFFF7604H	DSA13L	DMA source address register LCH13				√		0000H
FFFF7606H	DSA13H	DMA source address register HCH13				√		0000H
FFFF7608H	DSC13	DMA source chip select register CH13				√		0001H
FFFF760CH	DNSA13	DMA next source address register CH13					√	00000000H
FFFF760CH	DNSA13L	DMA next source address register LCH13				√		0000H
FFFF760EH	DNSA13H	DMA next source address register HCH13				√		0000H
FFFF7610H	DNSC13	DMA next source chip select register CH13				√		0001H
FFFF7614H	DDA13	DMA destination address register CH13					√	00000000H
FFFF7614H	DDA13L	DMA destination address register LCH13				√		0000H
FFFF7616H	DDA13H	DMA destination address register HCH13				√		0000H
FFFF7618H	DDC13	DMA destination chip select register CH13				√		0001H

(10/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF761CH	DNDA13	DMA next destination address register CH13	R/W				√	00000000H
FFFF761CH	DNDA13L	DMA next destination address register LCH13				√		0000H
FFFF761EH	DNDA13H	DMA next destination address register HCH13				√		0000H
FFFF7620H	DNDC13	DMA next destination chip select register CH13				√		0001H
FFFF7622H	DTC13	DMA transfer count register CH13				√		0000H
FFFF7624H	DNTC13	DMA next transfer count register CH13				√		0000H
FFFF7626H	DTCC13	DMA transfer count compare register CH13				√		0000H
FFFF7628H	DTCT13	DMA transfer control register CH13				√		0000H
FFFF762AH	DTS13	DMA transfer status register CH13		√	√			00H
FFFF7630H	DTRS14	DMA transfer request selection register CH14				√		0000H
FFFF7634H	DSA14	DMA source address register CH14					√	00000000H
FFFF7634H	DSA14L	DMA source address register LCH14				√		0000H
FFFF7636H	DSA14H	DMA source address register HCH14				√		0000H
FFFF7638H	DSC14	DMA source chip select register CH14				√		0001H
FFFF763CH	DNSA14	DMA next source address register CH14					√	00000000H
FFFF763CH	DNSA14L	DMA next source address register LCH14				√		0000H
FFFF763EH	DNSA14H	DMA next source address register HCH14				√		0000H
FFFF7640H	DNSC14	DMA next source chip select register CH14				√		0001H
FFFF7644H	DDA14	DMA destination address register CH14					√	00000000H
FFFF7644H	DDA14L	DMA destination address register LCH14				√		0000H
FFFF7646H	DDA14H	DMA destination address register HCH14				√		0000H
FFFF7648H	DDC14	DMA destination chip select register CH14				√		0001H
FFFF764CH	DNDA14	DMA next destination address register CH14					√	00000000H
FFFF764CH	DNDA14L	DMA next destination address register LCH14				√		0000H
FFFF764EH	DNDA14H	DMA next destination address register HCH14				√		0000H
FFFF7650H	DNDC14	DMA next destination chip select register CH14				√		0001H
FFFF7652H	DTC14	DMA transfer count register CH14				√		0000H
FFFF7654H	DNTC14	DMA next transfer count register CH14				√		0000H
FFFF7656H	DTCC14	DMA transfer count compare register CH14				√		0000H
FFFF7658H	DTCT14	DMA transfer control register CH14				√		0000H
FFFF765AH	DTS14	DMA transfer status register CH14		√	√			00H
FFFF7660H	DTRS15	DMA transfer request selection register CH15				√		0000H

(11/11)

Address	Symbol	Function register name	R/W	Access bit unit				Initial value
				1	8	16	32	
FFFF7664H	DSA15	DMA source address register CH15	R/W				√	00000000H
FFFF7664H	DSA15L	DMA source address register LCH15				√		0000H
FFFF7666H	DSA15H	DMA source address register HCH15				√		0000H
FFFF7668H	DSC15	DMA source chip select register CH15				√		0001H
FFFF766CH	DNSA15	DMA next source address register CH15					√	00000000H
FFFF766CH	DNSA15L	DMA next source address register LCH15				√		0000H
FFFF766EH	DNSA15H	DMA next source address register HCH15				√		0000H
FFFF7670H	DNSC15	DMA next source chip select register CH15				√		0001H
FFFF7674H	DDA15	DMA destination address register CH15					√	00000000H
FFFF7674H	DDA15L	DMA destination address register LCH15				√		0000H
FFFF7676H	DDA15H	DMA destination address register HCH15				√		0000H
FFFF7678H	DDC15	DMA destination chip select register CH15				√		0001H
FFFF767CH	DNDA15	DMA next destination address register CH15					√	00000000H
FFFF767CH	DNDA15L	DMA next destination address register LCH15				√		0000H
FFFF767EH	DNDA15H	DMA next destination address register HCH15				√		0000H
FFFF7680H	DNDC15	DMA next destination chip select register CH15				√		0001H
FFFF7682H	DTC15	DMA transfer count register CH15				√		0000H
FFFF7684H	DNTC15	DMA next transfer count register CH15				√		0000H
FFFF7686H	DTCC15	DMA transfer count compare register CH15				√		0000H
FFFF7688H	DTCT15	DMA transfer control register CH15				√		0000H
FFFF768AH	DTS15	DMA transfer status register CH15		√	√			00H

Caution If an unmapped address is accessed, a write access is ignored and 0 is returned in response to a read access.

6.4.3 Enabling or disabling writing control registers

The following control registers cannot be written while DMA transfers are enabled. However, all these registers can always be read.

Table 6-7 Whether writing to control registers is enabled or disabled

Always writable	DTRCx, DNSAnL, DNSAnH, DNSCn, DNDAnL, DNDAnH, DNDCn, DNTCn, DTSn
Writing prohibited while DMA transfers are enabled (= 1) (Operation is not guaranteed if these registers are written.)	DTRSn, DSAnL, DSAnH, DSCn, DDAnL, DDAnH, DDCn, DTCn, DTCCn, DTCTn

n = 0 to 15, x = 0 or 1

6.5 DMAC Control Registers

6.5.1 DTRCx - DMA transfer request control register (x = 0, 1)

Access This register can be read or written in 8-bit or 1-bit units.

Address DTRC0: $FFFF7300_H$, DTRC1: $FFFF7500_H$

Initial value 00_H

7	6	5	4	3	2	1	0
DTRCx ERR	0	0	0	0	0	0	DTRCx ADS
R/W	R	R	R	R	R	R	R/W

Table 6-8 DTRCx register contents

Bit position	Bit name	Function
7	DTRCxERR	DMA transfer error status This bit indicates that an error response has been received from the transfer target during a DMA transfer. If an error response is received, the DTRCxERR and DTRCxADS bits are set and a SysError exception is generated by the CPU. To clear this bit, write 0 to it. 0: No DMA transfer error 1: DMA transfer error
0	DTRCxADS	DMA transfer suspended This bit indicates that a DMA transfer has been suspended by a transfer stop request. In addition, the current DMA transfer can be suspended if the user writes 1 to this bit. 0: DMA transfer not suspended 1: DMA transfer suspended/DMA transfer abort request

6.5.2 DTRSn - DMA transfer request selection register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DTRS15: FFFF7660$_H$, DTRS14: FFFF7630$_H$, DTRS13: FFFF7600$_H$, DTRS12: FFFF75D0$_H$, DTRS11: FFFF75A0$_H$, DTRS10: FFFF7570$_H$, DTRS9: FFFF7540$_H$, DTRS8: FFFF7510$_H$, DTRS7: FFFF7460$_H$, DTRS6: FFFF7430$_H$, DTRS5: FFFF7400$_H$, DTRS4: FFFF73D0$_H$, DTRS3: FFFF73A0$_H$, DTRS2: FFFF7370$_H$, DTRS1: FFFF7340$_H$, DTRS0: FFFF7310$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	DTR3	DTR2	DTR1	DTR0
R	R	R	R	R/W	R/W	R/W	R/W

Table 6-9 DTRSn register contents

Bit position	Bit name	Function
3 to 0	DTR3 to DTR0	DMA transfer request assignment Specify assignment of a DMA transfer request to channel n.

DTR3	DTR2	DTR1	DTR0	DMA transfer request
0	0	0	0	Software DMA transfer request
0	0	0	1	Hardware DMA transfer request
Other than above				Setting prohibited

Cautions

1. Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1).If they are written, the operation is not guaranteed.
2. Operation is not guaranteed if the DTR[3:0] bits are set to a setting-prohibited value.

6.5.3 DSAnL - DMA source address register L (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DSA15L: FFFF7664$_H$, DSA14L: FFFF7634$_H$, DSA13L: FFFF7604$_H$, DSA12L: FFFF75D4$_H$, DSA11L: FFFF75A4$_H$, DSA10L: FFFF7574$_H$, DSA9L: FFFF7544$_H$, DSA8L: FFFF7514$_H$, DSA7L: FFFF7464$_H$, DSA6L: FFFF7434$_H$, DSA5L: FFFF7404$_H$, DSA4L: FFFF73D4$_H$, DSA3L: FFFF73A4$_H$, DSA2L: FFFF7374$_H$, DSA1L: FFFF7344$_H$, DSA0L: FFFF7314$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-10 DSAnL register contents

Bit position	Bit name	Function
15 to 0	SA15 to SA0	DMA source address Specify the lower 16 bits of the transfer source address on channel n. If this register is referenced during a DMA transfer, the address from which data is to be transferred next can be read. When referencing this register, it is recommended to access this register together with the DSAnL and DSAnH registers in 32-bit units. If the DNSAnH.NSAV bit is not set to 1 when a DMA transfer has been completed, the values of these bits return to the values when the DMA transfer was started.

Cautions

1. Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.
2. Specify an address by accessing in 32-bit units while the DTSnDTE bit is 0 to avoid data being transferred from an address that has not been completely specified.
3. DMA transfers of misaligned data is not supported. The lower 4 bits of an address corresponding to the transfer data size are as below (x indicates any bit).

   The operation is not guaranteed if a setting other than the following is specified.

Data size	SA3	SA2	SA1	SA0
8 bits	x	x	x	x
16 bits	x	x	x	0
32 bits	x	x	0	0
128 bits	0	0	0	0

4. Reading this register together with the DSAn register in 32-bit units while DMA transfers are enabled (DTESn.DTSnDTE = 1), might return an undefined value. Whether the read value is correct or incorrect can be judged as follows:

Judgment method:

- Read the DSAn register in 32-bit units twice in a row (first read → second read).

  If the higher 16 bits of the value read first are the same as the higher 16 bits of the value read second, the value read second is correct.

  If the higher 16 bits of the value read first differ from the higher 16 bits of the value read second, the value read first is correct.

6.5.4 DSAnH - DMA source address register H (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DSA15H: FFFF7666$_H$, DSA14H: FFFF7636$_H$, DSA13H: FFFF7606$_H$, DSA12H: FFFF75D6$_H$, DSA11H: FFFF75A6$_H$, DSA10H: FFFF7576$_H$, DSA9H: FFFF7546$_H$, DSA8H: FFFF7516$_H$, DSA7H: FFFF7466$_H$, DSA6H: FFFF7436$_H$, DSA5H: FFFF7406$_H$, DSA4H: FFFF73D6$_H$, DSA3H: FFFF73A6$_H$, DSA2H: FFFF7376$_H$, DSA1H: FFFF7346$_H$, DSA0H: FFFF7316$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
0	0	0	SA28	SA27	SA26	SA25	SA24
R	R	R	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-11 DSAnH register contents

Bit position	Bit name	Function
12 to 0	SA28 to SA16	DMA source address Specify the lower 13 bits of the transfer source address on channel n. If this register is referenced during a DMA transfer, the address from which data is to be transferred next can be read. When referencing this register, it is recommended to access this register together with the DSAnL and DSAnH registers in 32-bit units. If the DNSAnH.NSAV bit is not set to 1 when the DMA transfer has been completed, the values of these bits return to the values when the DMA transfer was started.

Cautions

1. Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.
2. Specify an address by accessing in 32-bit units while the DTSnDTE bit is 0 to avoid data being transferred from an address that has not been completely specified.
3. Reading this register together with the DSAn register in 32-bit units while DMA transfers are enabled (DTESn.DTSnDTE = 1), might return an undefined value. Whether the read value is correct or incorrect can be judged as follows:

   Judgment method:

   - Read the DSAn register in 32-bit units twice in a row (first read → second read).

     If the higher 16 bits of the value read first are the same as the higher 16 bits of the value read second, the value read second is correct.

     If the higher 16 bits of the value read first differ from the higher 16 bits of the value read second, the value read first is correct.

6.5.5 DSCn - DMA source chip select register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DSC15: FFFF7668$_H$, DSC14: FFFF7638$_H$, DSC13: FFFF7608$_H$, DSC12: FFFF75D8$_H$, DSC11: FFFF75A8$_H$, DSC10: FFFF7578$_H$, DSC9: FFFF7548$_H$, DSC8: FFFF7518$_H$, DSC7: FFFF7468$_H$, DSC6: FFFF7438$_H$, DSC5: FFFF7408$_H$, DSC4: FFFF73D8$_H$, DSC3: FFFF73A8$_H$, DSC2: FFFF7378$_H$, DSC1: FFFF7348$_H$, DSC0: FFFF7318$_H$

Initial value 0001$_H$

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	SCS1	SCS0	SCSE
R	R	R	R	R	R/W	R/W	R/W

Table 6-12 DSCn register contents

Bit position	Bit name	Function
2 to 0	SCS1, SCS0, SESE	DMA source chip select Specify an area to be selected as the transfer source on channel n.

SCS1	SCS0	SCSE	Data flash
0	0	1	Data flash, external memory area, peripheral I/O area, and HBUS-RAM
0	1	0	Code flash and local RAM
Other than above			Setting prohibited

Cautions

1. Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.
2. Specify the SCS0 and SCSE bits so that only one of them is 1. If both of these bits are set to 1, the operation is not guaranteed.
3. Be sure to clear the SCS1 bit to 0.

6.5.6 DNSAnL - DMA next source address register L (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DNSA15L: FFFF766C$_H$, DNSA14L: FFFF763C$_H$, DNSA13L: FFFF760C$_H$, DNSA12L: FFFF75DC$_H$, DNSA11L: FFFF75AC$_H$, DNSA10L: FFFF757C$_H$, DNSA9L: FFFF754C$_H$, DNSA8L: FFFF751C$_H$, DNSA7L: FFFF746C$_H$, DNSA6L: FFFF743C$_H$, DNSA5L: FFFF740C$_H$, DNSA4L: FFFF73DC$_H$, DNSA3L: FFFF73AC$_H$, DNSA2L: FFFF737C$_H$, DNSA1L: FFFF734C$_H$, DNSA0L: FFFF731C$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
NSA15	NSA14	NSA13	NSA12	NSA11	NSA10	NSA9	NSA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
NSA7	NSA6	NSA5	NSA4	NSA3	NSA2	NSA1	NSA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-13 DNSAnL register contents

Bit position	Bit name	Function
15 to 0	NSA15 to NSA0	DMA next source address Specify the lower 16 bits of the transfer source address on channel n.

Caution DMA transfers of misaligned data is not supported. The lower 4 bits of an address corresponding to the transfer data size are as below (x indicates any bit).

The operation is not guaranteed if a setting other than the following is specified.

Data size	NSA3	NSA2	NSA1	NSA0
8 bits	x	x	x	x
16 bits	x	x	x	0
32 bits	x	x	0	0
128 bits	0	0	0	0

6.5.7 DNSAnH - DMA next source address register H (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DNSA15H: FFFF766E$_H$, DNSA14H: FFFF763E$_H$, DNSA13H: FFFF760E$_H$, DNSA12H: FFFF75DE$_H$, DNSA11H: FFFF75AE$_H$, DNSA10H: FFFF757E$_H$, DNSA9H: FFFF754E$_H$, DNSA8H: FFFF751E$_H$, DNSA7H: FFFF746E$_H$, DNSA6H: FFFF743E$_H$, DNSA5H: FFFF740E$_H$, DNSA4H:FFFF73DE$_H$, DNSA3H: FFFF73AE$_H$, DNSA2H: FFFF737E$_H$, DNSA1H: FFFF734E$_H$, DNSA0H: FFFF731E$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
NSAV	0	0	NSA28	NSA27	NSA26	NSA25	NSA24
R/W	R	R	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
NSA23	NSA22	NSA21	NSA20	NSA19	NSA18	NSA17	NSA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-14 DNSAnH register contents

Bit position	Bit name	Function
15	NSAV	DMA next source address valid Specify whether to copy a chip select signal from the DMA next source address register to the DMA source address register when the DMA transfer has been completed. It is cleared when the address has been copied. 0: Does not copy/copying completed 1: Copies/copying not completed
12 to 0	NSA28 to NSA16	DMA next source address Specify the lower 13 bits of the transfer source address on channel n.

6.5.8 DNSCn - DMA next source chip select register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DNSC15: $FFFF7670_H$, DNSC14: $FFFF7640_H$, DNSC13: $FFFF7610_H$, DNSC12: $FFFF75E0_H$, DNSC11: $FFFF75B0_H$, DNSC10: $FFFF7580_H$, DNSC9: $FFFF7550_H$, DNSC8: $FFFF7520_H$, DNSC7: $FFFF7470_H$, DNSC6: $FFFF7440_H$, DNSC5: $FFFF7410_H$, DNSC4: $FFFF73E0_H$, DNSC3: $FFFF73B0_H$, DNSC2: $FFFF7380_H$, DNSC1: $FFFF7350_H$, DNSC0: $FFFF7320_H$

Initial value 0001_H

15	14	13	12	11	10	9	8
NSCV	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	NSCS1	NSCS0	NSCSE
R	R	R	R	R	R/W	R/W	R/W

Table 6-15 DNSCn register contents

Bit position	Bit name	Function
15	NSCV	DMA next source address select valid Specify whether to copy a chip select signal from the DMA next source chip select register to the DMA source chip select register when the DMA transfer has been completed. It is cleared when the chip select signal has been copied. 0: Does not copy/copying completed 1: Copies/copying not completed
2 to 0	NSCS1, NSCS0, NSCSE	DMA next source chip select Specify an area to be selected as the transfer source on channel n.

NSCS1	NSCS0	NSCSE	Selected area
0	0	1	Data flash, external memory area, peripheral I/O area, and HBUS-RAM
0	1	0	Code flash and local RAM
Other than above			Setting prohibited

Cautions
1. Specify the NSCS0 and NSCSE bits so that only one of them is 1. If both of these bits are set to 1, the operation is not guaranteed.
2. Be sure to clear the NSCS bit to 0.

6.5.9 DDAnL - DMA destination address register L (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DDA15L: FFFF7674$_H$, DDA14L: FFFF7644$_H$, DDA13L: FFFF7614$_H$, DDA12L: FFFF75E4$_H$, DDA11L: FFFF75B4$_H$, DDA10L: FFFF7584$_H$, DDA9L: FFFF7554$_H$, DDA8L: FFFF7524$_H$, DDA7L: FFFF7474$_H$, DDA6L: FFFF7444$_H$, DDA5L: FFFF7414$_H$, DDA4L: FFFF73E4$_H$, DDA3L: FFFF73B4$_H$, DDA2L: FFFF7384$_H$, DDA1L: FFFF7354$_H$, DDA0L: FFFF7324$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-16 DDAnL register contents

Bit position	Bit name	Function
15 to 0	DA15 to DA0	DMA destination address Specify the lower 16 bits of the transfer source address on channel n. If this register is referenced during a DMA transfer, the address from which data is to be transferred next can be read. When referencing this register, it is recommended to access this register together with the DSAnL and DDAnH register in 32-bit units. If the DNDAnH.NDAV bit is not set to 1 when the DMA transfer has been completed, the values of these bits return to the values when the DMA transfer was started.

Cautions

1. Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.
2. Specify an address by accessing in 32-bit units while the DTSnDTE bit is 0 to avoid data being transferred from an address that has not been completely specified.
3. If an error occurs in the transfer target in the read cycle of a DMA transfer, the write cycle is not executed but the destination address is updated.
4. DMA transfers of misaligned data is not supported. The lower 4 bits of an address corresponding to the transfer data size are as below (x indicates any bit). The operation is not guaranteed if a setting other than the following is specified.

Data size	DA3	DA2	DA1	DA0
8 bits	x	x	x	x
16 bits	x	x	x	0
32 bits	x	x	0	0
128 bits	0	0	0	0

5. Reading this register together with the DDAn register in 32-bit units while DMA transfers are enabled (DTESn.DTSnDTE = 1), might return an undefined value. Whether the read value is correct or incorrect can be judged as follows:
   Judgment method:
   - Read the DDAn register in 32-bit units twice in a row (first read → second read).

     If the higher 16 bits of the value read first are the same as the higher 16 bits of the value read second, the value read second is correct.

     If the higher 16 bits of the value read first differ from the higher 16 bits of the value read second, the value read first is correct.

6.5.10 DDAnH - DMA destination address register H (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DDA15H: $FFFF7676_H$, DDA14H: $FFFF7646_H$, DDA13H: $FFFF7616_H$, DDA12H: $FFFF75E6_H$, DDA11H: $FFFF75B6_H$, DDA10H: $FFFF7586_H$, DDA9H: $FFFF7556_H$, DDA8H: $FFFF7526_H$, DDA7H: $FFFF7476_H$, DDA6H: $FFFF7446_H$, DDA5H: $FFFF7416_H$, DDA4H: $FFFF73E6_H$, DDA3H: $FFFF73B6_H$, DDA2H: $FFFF7386_H$, DDA1H: $FFFF7356_H$, DDA0H: $FFFF7326_H$

Initial value 0000_H

15	14	13	12	11	10	9	8
0	0	0	DA28	DA27	DA26	DA25	DA24
R	R	R	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-17 DDAnH register contents

Bit position	Bit name	Function
12 to 0	DA28 to DA16	DMA destination address Specify the lower 13 bits of the transfer source address on channel n. If this register is referenced during a DMA transfer, the address from which data is to be transferred next can be read. When referencing this register, it is recommended to access this register together with the DSAnL and DDAnH register in 32-bit units. If the DNDAnH.NDAV bit is not set to 1 when the DMA transfer has been completed, the values of these bits return to the values when the DMA transfer was started.

Cautions

1. Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.
2. Specify an address by accessing in 32-bit units while the DTSnDTE bit is 0 to avoid data being transferred from an address that has not been completely specified.
3. If an error occurs in the transfer target in the read cycle of a DMA transfer, the write cycle is not executed but the destination address is updated.
4. Reading this register together with the DDAn register in 32-bit units while DMA transfers are enabled (DTESn.DTSnDTE = 1), might return an undefined value. Whether the read value is correct or incorrect can be judged as follows:
   Judgment method:
   - Read the DDAn register in 32-bit units twice in a row (first read → second read).

     If the higher 16 bits of the value read first are the same as the higher 16 bits of the value read second, the value read second is correct.

     If the higher 16 bits of the value read first differ from the higher 16 bits of the value read second, the value read first is correct.

6.5.11 DDCn - DMA destination chip select register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DDC15: $FFFF7678_H$, DDC14: $FFFF7648_H$, DDC13: $FFFF7618_H$, DDC12: $FFFF75E8_H$, DDC11: $FFFF75B8_H$, DDC10: $FFFF7588_H$, DDC9: $FFFF7558_H$, DDC8: $FFFF7528_H$, DDC7: $FFFF7478_H$, DDC6: $FFFF7448_H$, DDC5: $FFFF7418_H$, DDC4: $FFFF73E8_H$, DDC3: $FFFF73B8_H$, DDC2: $FFFF7388_H$, DDC1: $FFFF7358_H$, DDC0: $FFFF7328_H$

Initial value 0001_H

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	DCS1	DCS0	DCSE
R	R	R	R	R	R/W	R/W	R/W

Table 6-18 DDCn register contents

Bit position	Bit name	Function
2 to 0	DCS1, DCS0, DCSE	DMA destination chip select Specify an area to be selected as the transfer source on channel n.

<R>

DCS1	DCS0	DCSE	Selected area
0	0	1	External memory area, peripheral I/O area, and HBUS-RAM
0	1	0	Local RAM
Other than above			Setting prohibited

<R>

Cautions

1. Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.
2. Specify the DCS0 and DCSE bits so that only one of them is 1. If both of these bits are set to 1, the operation is not guaranteed.
3. Be sure to clear the DCS1 bit to 0.

6.5.12 DNDAnL - DMA next destination address register L (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DNDA15L: FFFF767C$_H$, DNDA14L: FFFF764C$_H$, DNDA13L: FFFF761C$_H$, DNDA12L: FFFF75EC$_H$, DNDA11L: FFFF75BC$_H$, DNDA10L: FFFF758C$_H$, DNDA9L: FFFF755C$_H$, DNDA8L: FFFF752C$_H$, DNDA7L: FFFF747C$_H$, DNDA6L: FFFF744C$_H$, DNDA5L: FFFF741C$_H$, DNDA4L: FFFF73EC$_H$, DNDA3L: FFFF73BC$_H$, DNDA2L: FFFF738C$_H$, DNDA1L: FFFF735C$_H$, DNDA0L: FFFF732C$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
NDA15	NDA14	NDA13	NDA12	NDA11	NDA10	NDA9	NDA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
NDA7	NDA6	NDA5	NDA4	NDA3	NDA2	NDA1	NDA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-19 DNDAnL register contents

Bit position	Bit name	Function
15 to 0	NDA15 to NDA0	DMA next destination address Specify the lower 16 bits of the transfer source address on channel n.

Caution DMA transfers of misaligned data is not supported. The lower 4 bits of an address corresponding to the transfer data size are as below (x indicates any bit).

The operation is not guaranteed if a setting other than the following is specified.

Data size	NDA3	NDA2	NDA1	NDA0
8 bits	x	x	x	x
16 bits	x	x	x	0
32 bits	x	x	0	0
128 bits	0	0	0	0

6.5.13 DNDAnH - DMA next destination address register H (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DNDA15H: FFFF767E$_H$, DNDA14H: FFFF764E$_H$, DNDA13H: FFFF761E$_H$, DNDA12H: FFFF75EE$_H$, DNDA11H: FFFF75BE$_H$, DNDA10H: FFFF758E$_H$, DNDA9H: FFFF755E$_H$, DNDA8H: FFFF752E$_H$, DNDA7H: FFFF747E$_H$, DNDA6H: FFFF744E$_H$, DNDA5H: FFFF741E$_H$, DNDA4H: FFFF73EE$_H$, DNDA3H: FFFF73BE$_H$, DNDA2H: FFFF738E$_H$, DNDA1H: FFFF735E$_H$, DNDA0H: FFFF732E$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
NDAV	0	0	NDA28	NDA27	NDA26	NDA25	NDA24
R/W	R	R	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
NDA23	NDA22	NDA21	NDA20	NDA19	NDA18	NDA17	NDA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-20 DNDAnH register contents

Bit position	Bit name	Function
15	NDAV	DMA next destination address valid Specify whether to copy a chip select signal from the DMA next source chip select register to the DMA source chip select register when the DMA transfer has been completed. It is cleared when the chip select signal has been copied. 0: Does not copy/copying completed 1: Copies/copying not completed
12 to 0	NDA28 to NDA16	DMA next destination address Specify the lower 13 bits of the transfer source address on channel n.

6.5.14 DNDCn - DMA next destination chip select register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DNDC15: $FFFF7680_H$, DNDC14: $FFFF7650_H$, DNDC13: $FFFF7620_H$, DNDC12: $FFFF75F0_H$, DNDC11: $FFFF75C0_H$, DNDC10: $FFFF7590_H$, DNDC9: $FFFF7560_H$, DNDC8: $FFFF7530_H$, DNDC7: $FFFF7480_H$, DNDC6: $FFFF7450_H$, DNDC5: $FFFF7420_H$, DNDC4: $FFFF73F0_H$, DNDC3: $FFFF73C0_H$, DNDC2: $FFFF7390_H$, DNDC1: $FFFF7360_H$, DNDC0: $FFFF7330_H$

Initial value 0001_H

15	14	13	12	11	10	9	8
NDCV	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	NDCS1	NDCS0	NDCSE
R	R	R	R	R	R/W	R/W	R/W

Table 6-21 DNDCn register contents

Bit position	Bit name	Function
15	NDCV	DMA next destination address valid Specify whether to copy a chip select signal from the DMA next destination chip select register to the DMA source chip select register when the DMA transfer has been completed.It is cleared when the chip select signal has been copied. 0: Does not copy/copying completed 1: Copies/copying not completed
2 to 0	NDCS1, NDCS0, NDCSE	DMA next destination chip select Specify an area to be selected as the transfer source on channel n.

NDCS1	NDCS0	NDCSE	Selected area
0	0	1	Data flash, external memory area, peripheral I/O area, and HBUS-RAM
0	1	0	Code flash and local RAM
Other than above			Setting prohibited

Cautions
1. Specify the NDCS0 and NDCSE bits so that only one of them is 1.If both of these bits are set to 1, the operation is not guaranteed.
2. Be sure to clear the NDCS1 bit to 0.

6.5.15 DTCn - DMA transfer count register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DCT15: $FFFF7682_H$, DCT14: $FFFF7652_H$, DCT13: $FFFF7622_H$, DCT12: $FFFF75F2_H$, DCT11: $FFFF75C2_H$, DCT10: $FFFF7592_H$, DCT9: $FFFF7562_H$, DCT8: $FFFF7532_H$, DCT7: $FFFF7482_H$, DCT6: $FFFF7452_H$, DCT5: $FFFF7422_H$, DCT4: $FFFF73F2_H$, DCT3: $FFFF73C2_H$, DCT2: $FFFF7392_H$, DCT1: $FFFF7362_H$, DCT0: $FFFF7332_H$

Initial value 0000_H

15	14	13	12	11	10	9	8
0	DTC14	DTC13	DTC12	DTC11	DTC10	DTC9	DTC8
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DTC7	DTC6	DTC5	DTC4	DTC3	DTC2	DTC1	DTC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-22 DTCn register contents

Bit position	Bit name	Function
14 to 0	DTC14 to DTC0	DMA transfer count Specify the number of times of DMA transfers (DMA transfer count) on channel n. When this register is referenced during a DMA transfer, the remaining number of times DMA transfer to be executed can be read. If the DNTCn.NTCV bit is not set to 1, these bits hold the values when the DMA transfer has been completed (0000H).

DTC[14:0]	Operation
0000H	Transfer executed 32,768 times or until completion of transfer
0001H	Transfer executed once or transfer to be executed once
...	...
7FFFH	Transfer executed 32,767 times or 32,767 times of transfer to be executed

Cautions

1. Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.
2. If an error occurs in the transfer target in the read cycle of a DMA transfer, the write cycle is not executed but the destination address is updated.

6.5.16 DNTCn - DMA next transfer count register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DNCT15: $FFFF7684_H$, DNCT14: $FFFF7654_H$, DNCT13: $FFFF7624_H$, DNCT12: $FFFF75F4_H$, DNCT11: $FFFF75C4_H$, DNCT10: $FFFF7594_H$, DNCT9: $FFFF7564_H$, DNCT8: $FFFF7534_H$, DNCT7: $FFFF7484_H$, DNCT6: $FFFF7454_H$, DNCT5: $FFFF7424_H$, DNCT4: $FFFF73F4_H$, DNCT3: $FFFF73C4_H$, DNCT2: $FFFF7394_H$, DNCT1: $FFFF7364_H$, DNCT0: $FFFF7334_H$

Initial value 0000_H

15	14	13	12	11	10	9	8
NTCV	NDTC14	NDTC13	NDTC12	NDTC11	NDTC10	NDTC9	NDTC8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
NDTC7	NDTC6	NDTC5	NDTC4	NDTC3	NDTC2	NDTC1	NDTC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-23 DNTCn register contents

Bit position	Bit name	Function
15	NTCV	DMA next transfer count valid Specify whether to copy the DMA transfer count from the DMA next transfer count register to the DMA count register when the DMA transfer has been completed. It is cleared when the DMA transfer count has been copied. 0: Does not copy/copying completed 1: Copies/copying not completed
14 to 0	NDTC14 to NDTC0	DMA next transfer count Specify the number of times of DMA transfers (DMA transfer count) on channel n.

NDTC[14:0]	Operation
0000H	Transfer executed 32,768 times
0001H	Transfer executed once
...	...
7FFFH	Transfer executed 32,767 times

6.5.17 DTCCn - DMA transfer count compare register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DTCC15: $FFFF7686_H$, DTCC14: $FFFF7656_H$, DTCC13: $FFFF7626_H$, DTCC12: $FFFF75F6_H$, DTCC11: $FFFF75C6_H$, DTCC10: $FFFF7596_H$, DTCC9: $FFFF7566_H$, DTCC8: $FFFF7536_H$, DTCC7: $FFFF7486_H$, DTCC6: $FFFF7456_H$, DTCC5: $FFFF7426_H$, DTCC4: $FFFF73F6_H$, DTCC3: $FFFF73C6_H$, DTCC2: $FFFF7396_H$, DTCC1: $FFFF7366_H$, DTCC0: $FFFF7336_H$

Initial value 0000_H

15	14	13	12	11	10	9	8
0	DTCC14	DTCC13	DTCC12	DTCC11	DTCC10	DTCC9	DTCC8
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DTCC7	DTCC6	DTCC5	DTCC4	DTCC3	DTCC2	DTCC1	DTCC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-24 DTCCn register contents

Bit position	Bit name	Function
14 to 0	DTCC14 to DTCC0	DMA transfer count comparison Specify the DMA transfer count to be compared with the value of the DMA transfer count register (DTCn) on channel n. If the values of these registers match, an interrupt is generated. This interrupt is used as a trigger for setting the next address.

DTCC[14:0]	Operation
0000H	Not compared
0001H	An interrupt is generated when DTC is 0001H.
...	...
7FFFH	An interrupt is generated when DTC is 7FFFH.

Caution Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.

6.5.18 DTCTn - DMA transfer control register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DTCT15: $FFFF7688_H$, DTCT14: $FFFF7658_H$, DTCT13: $FFFF7628_H$, DTCT12: $FFFF75F8_H$, DTCT11: $FFFF75C8_H$, DTCT10: $FFFF7598_H$, DTCT9: $FFFF7568_H$, DTCT8: $FFFF7538_H$, DTCT7: $FFFF7488_H$, DTCT6: $FFFF7458_H$, DTCT5: $FFFF7428_H$, DTCT4: $FFFF73F8_H$, DTCT3: $FFFF73C8_H$, DTCT2: $FFFF7398_H$, DTCT1: $FFFF7368_H$, DTCT0: $FFFF7338_H$

Initial value 0000_H

15	14	13	12	11	10	9	8
0	DS1	DS0	MLE	INF	0	0	0
R	R/W	R/W	R/W	R/W	R	R	R
7	6	5	4	3	2	1	0
SACM1	SACM0	DACM1	DACM0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R	R/W

Table 6-25 DTCTn register contents (1/2)

Bit position	Bit name	Function
14, 13	DS1, DS0	DMA transfer data size Specify the number of times of DMA transfers on channel n. DS1 / DS0 / Transfer data size: 0 / 0 / 8 bits 0 / 1 / 16 bits 1 / 0 / 32 bits 1 / 1 / 128 bits
12	MLE	Multi-link enable Specify whether to acknowledge the next DMA transfer request, even if the DTSnTC bit is not cleared to 0 after DMA transfer has been completed. If this bit is set to 1, the DTSn.DTE bit is not cleared when the DMA transfer finishes. Even if the TC bit is not cleared, the DMA transfer is executed if a DMA transfer request is issued. 0: Clear the DTSn.DTE bit when the DMA transfer finishes. 1: Do not clear the DTSn.DTE bit when the DMA transfer finishes.
11	INF	When the INF bit is set to 1, the valid bits (the DNSAnH.NSAV, DNSAn.NSCV, DNDAnH.NDAV, and DNDCn.NDCV bits) of the next register are not cleared even if the transfer direction, the transfer source, and the transfer frequency are copied from the DMA next register to the DMA current register.

Table 6-25 DTCTn register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7, 6</td><td>SACM1,
SACM0</td><td>DMA transfer destination address counting direction
Specify the direction in which the transfer destination address on channel n is to be counted.
<table>
<tr><th>SACM1</th><th>SACM0</th><th>Counting direction</th></tr>
<tr><td>0</td><td>0</td><td>Incremented</td></tr>
<tr><td>0</td><td>1</td><td>Decremented</td></tr>
<tr><td>1</td><td>0</td><td>Fixed</td></tr>
<tr><td>1</td><td>1</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>5, 4</td><td>DACM1,
DACM0</td><td>DMA transfer destination address counting direction
Specify the direction in which the transfer destination address on channel n is to be counted.
<table>
<tr><th>DACM1</th><th>DACM0</th><th>Counting direction</th></tr>
<tr><td>0</td><td>0</td><td>Incremented</td></tr>
<tr><td>0</td><td>1</td><td>Decremented</td></tr>
<tr><td>1</td><td>0</td><td>Fixed to 0</td></tr>
<tr><td>1</td><td>1</td><td>Setting prohibited</td></tr>
</table></td></tr>
</table>

Cautions

1. Writing these bits is prohibited while DMA transfers are enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.
2. The operation is not guaranteed if the SACM[1:0] and DACM[1:0] bits are set to a setting-prohibited value.
3. Be sure to clear bits 11 and 0 of the DTCTn register to 0.

6.5.19 DTSn - DMA transfer status register (n = 0 to 15)

Access This register can be read or written in 8-bit or 1-bit units.

Address DTS15: $FFFF768A_H$, DTS14: $FFFF765A_H$, DTS13: $FFFF762A_H$, DTS12: $FFFF75FA_H$, DTS11: $FFFF75CA_H$, DTS10: $FFFF759A_H$, DTS9: $FFFF756A_H$, DTS8: $FFFF753A_H$, DTS7: $FFFF748A_H$, DTS6: $FFFF745A_H$, DTS5: $FFFF742A_H$, DTS4: $FFFF73FA_H$, DTS3: $FFFF73CA_H$, DTS2: $FFFF739A_H$, DTS1: $FFFF736A_H$, DTS0: $FFFF733A_H$

Initial value 00_H

7	6	5	4	3	2	1	0
DTSnTC	DTSnDT	0	0	DTSnER	DTSnDR	DTSnSR	DTSnDTE
R/W	R/W	R	R	R	R	R/W	R/W

Table 6-26 DTSn register contents (1/2)

Bit position	Bit name	Function
7	DTSnTC	DMA transfer end status This bit indicates that the DMA transfer has been completed. Write 0 to this bit to clear it after reading 1 from it. It is recommended to write this bit using a bit manipulation instruction such as CLR1. 0: DMA transfer not completed 1: DMA transfer completed
6	DTSnDT	DMA transfer status This bit indicates that a DMA transfer request has been acknowledged and that the DMA transfer is in progress. It is not set to 1 when only a DMA transfer request is issued. This bit is cleared to 0 when the DMA transfer has been completed. If the DTSnDTE bit is 0, this bit can be cleared by the user. (It can also be written at the same time as the DTSnDTE bit.) 0: DMA transfer request acknowledged 1: DMA transfer in progress
3	DTSnER	DMA transfer error flag This bit indicates that a DMA transfer error has occurred on channel n. It is cleared to 0 when the DTRCx.DTRCxERR register is cleared. The DTSnER bit is read-only. 0: No DMA transfer error 1: DMA transfer error
2	DTSnDR	Hardware DMA transfer request This bit indicates that channel n has a hardware DMA transfer request. It is cleared to 0 when the hardware DMA transfer request is deasserted. This bit operates regardless of the status of the bit. It is not set to 1 by a software DMA transfer request, or by a hardware DMA transfer request when a software DMA transfer request is selected by the DMA transfer request select register.The DTSnDR bit is read-only. 0: No hardware DMA transfer request 1: No hardware DMA transfer request
1	DTSnSR	Software DMA transfer request This bit selects a software DMA transfer request. If a software DMA transfer request is selected by the DMA transfer request select register, writing 1 to the DTSnSR and DTE bits starts the DMA transfer. This bit is cleared to 0 when the DMA transfer has been completed. Writing 0 to DTSnSR bit aborts DMA transfer. 0: No hardware DMA transfer request 1: No hardware DMA transfer request

Table 6-26 DTSn register contents (2/2)

Bit position	Bit name	Function
0	DTSnDTE	DMA transfer enable Enable or disable DMA transfers. DMA transfer is executed if 1 is written to the DTSnDTE bit and a DMA transfer request is issued. This bit is cleared to 0 if the MLE bit is 0 when the DMA transfer has been completed. If 0 is written to the DTSnDTE bit during a DMA transfer, the DMA transfer is suspended. 0: Disable DMA transfers. 1: Enable DMA transfers.

6.6 DMAC Function Details

6.6.1 DMAC transfer setup flow

A DMAC transfer setup flow is shown below.

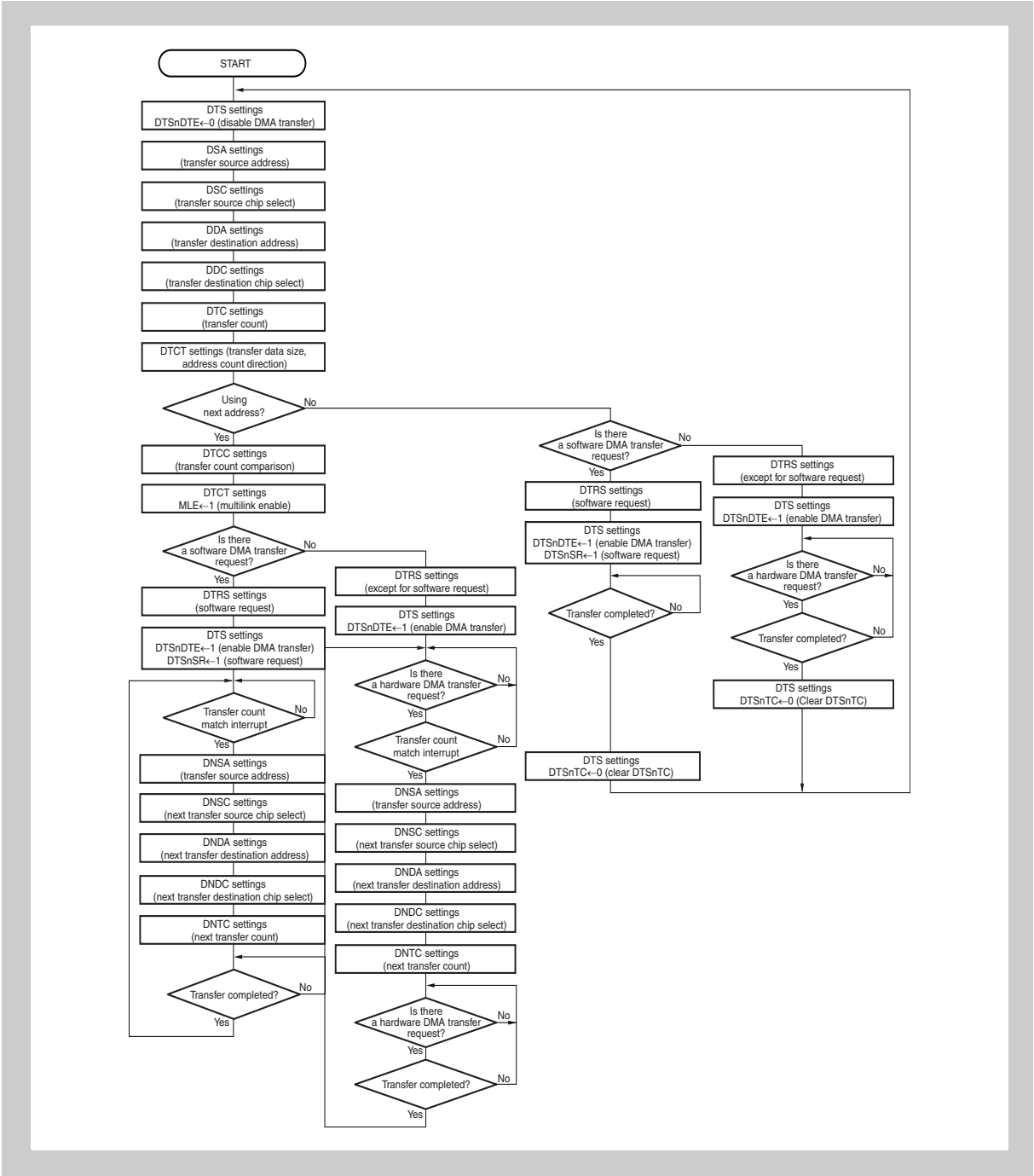


Figure 6-1 DMAC transfer setting flow

6.6.2 DMAC transfer modes

A single-transfer mode and a single-step transfer mode are supported as transfer modes.

In either mode, a transfer is executed in two cycles (dual address transfer) and therefore, a read cycle and a write cycle are generated each time a transfer is executed. For 128-bit transfers, the read cycle is generated four times and the write cycle is generated four times, in that order.

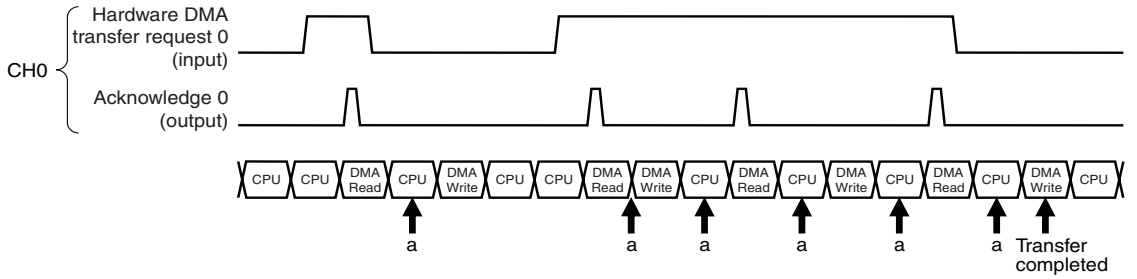
Note that the bus is not locked. Consequently, a CPU cycle might interrupt between the read and write cycles, and between the four read cycles and four write cycles during a 128-bit transfer.

(1) Single transfer mode (when a hardware DMA transfer request is generated)

When a hardware DMA transfer request is acknowledged, data of the transfer data size (8, 16, 32, or 128 bits) is transferred. Each time a transfer is executed, the bus is released and the system waits for a DMA transfer request.At this time, the acknowledge n signal that indicates that the hardware DMA transfer request has been acknowledged is also output (n = 15 to 0).

Each time a hardware DMA transfer request is acknowledged, the transfer is executed once. This operation is repeated the number of times specified by DMA transfer count register n (DTCn) (n = 15 to 0).

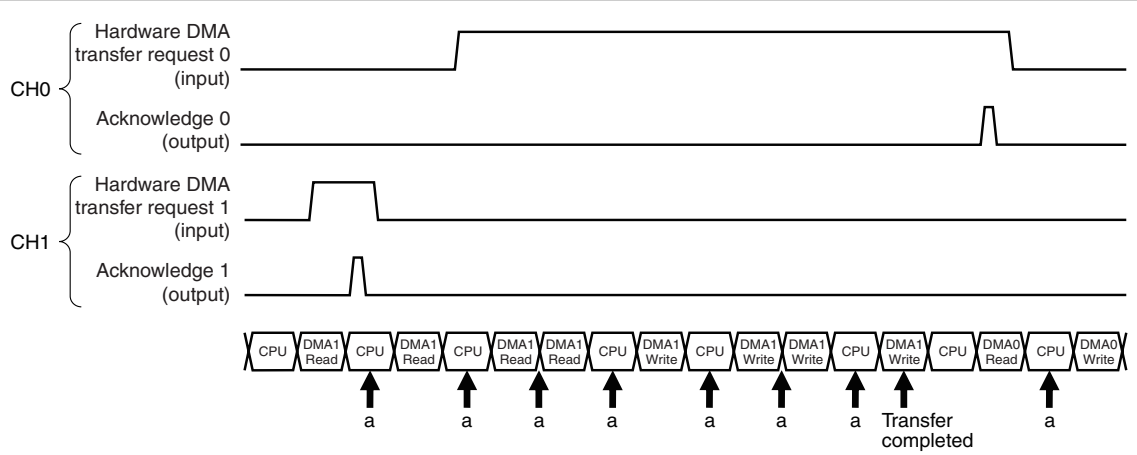
<R>



a) The bus is always released. When the CPU requests the bus mastership, the CPU acquires the bus mastership.

Figure 6-2 Example of single transfer (8/16/32 bits)

<R>



a) The bus is always released. When the CPU requests the bus mastership, the CPU acquires the bus mastership.

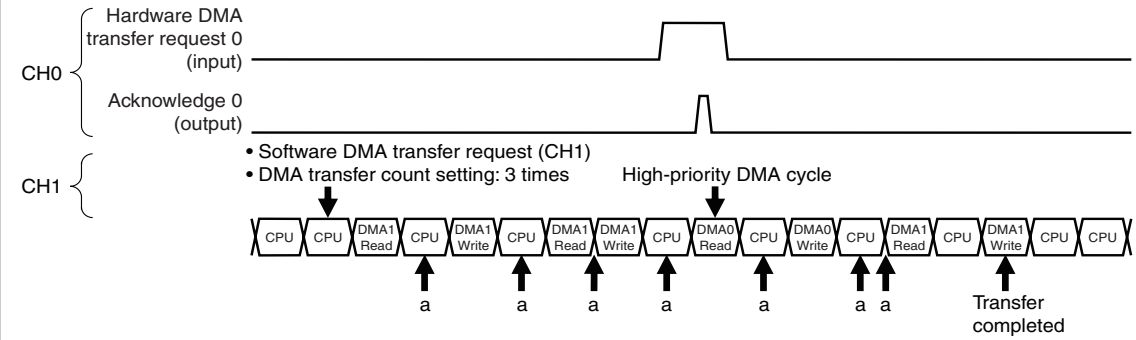
Figure 6-3 Example of single transfer (128 bits, DMA channel priority order: CHO (high) > CH1 (low))

(2) Single-step transfer mode (when a software DMA transfer request is generated)

When a hardware DMA transfer request is acknowledged, data of the transfer data size (8, 16, 32, or 128 bits) is transferred. Each time a transfer is executed, the bus is released and the system waits for a DMA transfer request. At this time, the acknowledge n signal that indicates that the hardware DMA transfer request has been acknowledged is also output (n = 15 to 0).

Each time a hardware DMA transfer request is acknowledged, the transfer is executed once. This operation is repeated the number of times specified by DMA transfer count register n (DTCn) (n = 15 to 0). Because the priority is identified each time a transfer is executed, the DMA cycle on a channel having the higher priority might interrupt.

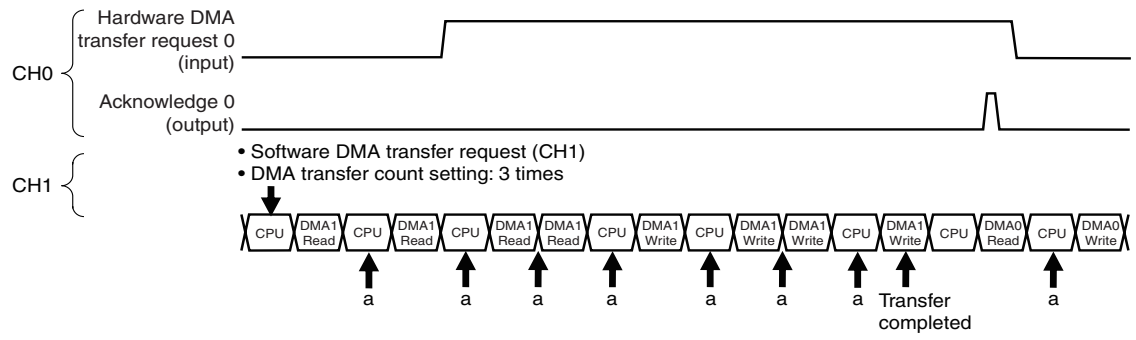
<R>



a) The bus is always released.When the CPU requests the bus mastership, the CPU acquires the bus mastership.

Figure 6-4 Example of single-step transfer (8 bits, DMA channel priority order: CHO (high) > CH1 (low))

<R>
<R>



a) The bus is always released.When the CPU requests the bus mastership, the CPU acquires the bus mastership.

Figure 6-5 Example of single-step transfer (128 bits, DMA channel priority order: CHO (high) > CH1 (low))

6.6.3 DMAC channel priority control

The priority on each channel is fixed and is as follows:

DMAC0 > DMAC1

DMAC0 (CH0 > CH1 > CH2 > CH3 > CH4 > CH5 > CH6 > CH7)

DMAC1 (CH8 > CH9 > CH10 > CH11 > CH12 > CH13 > CH14 > CH15)

If a DMA transfer request with a higher priority is generated, the DMA transfer request with the higher priority always takes precedence. When a software DMA transfer request is generated, the bus is also released each time a DMA cycle has been completed.

An example where a DMA transfer request with a higher priority is generated while a DMA transfer is being executed is shown below.

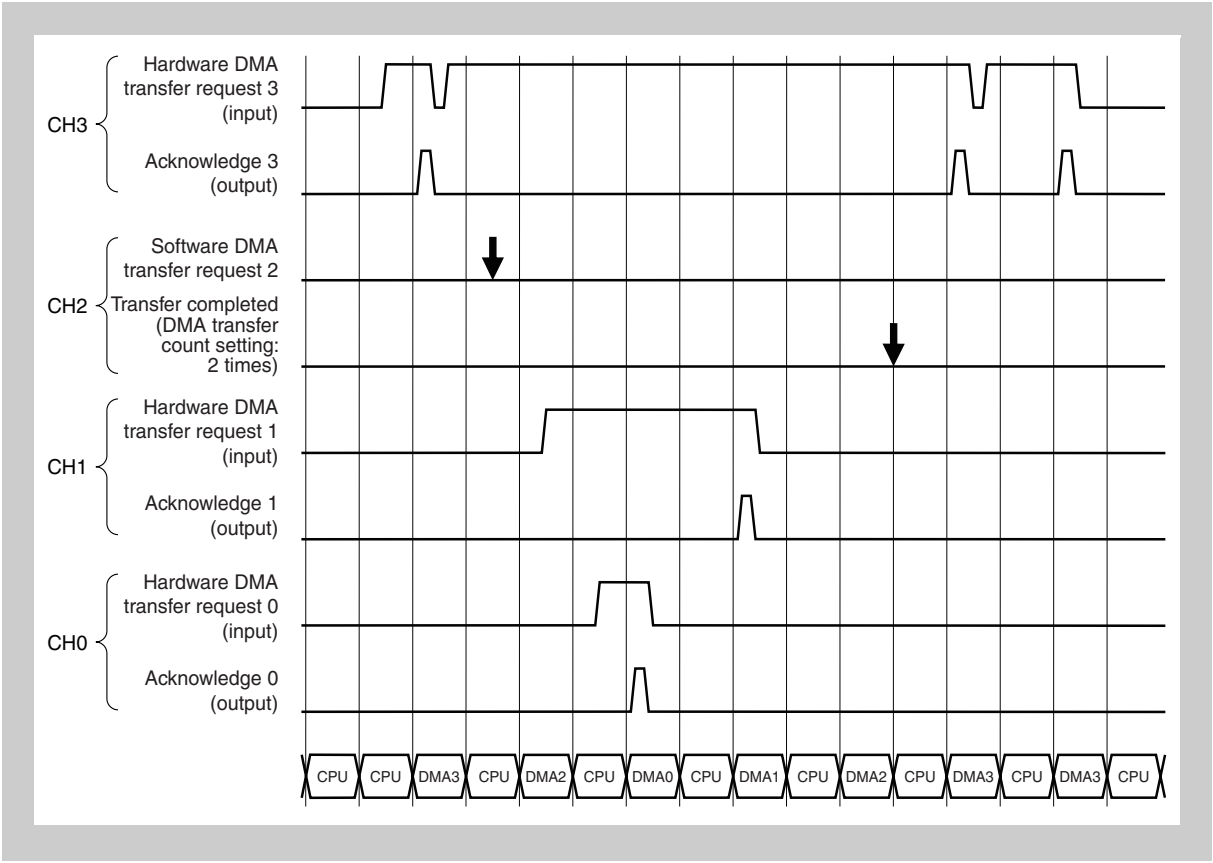


Figure 6-6 Example of priority control

6.6.4 Valid DMA transfer request conditions

Whether a DMA transfer request on channel n is acknowledged depends on the setting of the DTRCxERR and DTRCxADS bits of the DMA transfer request control register (DTRCx), the MLE bit of the DMA transfer control register (DTCTn), and the DTSnTC and DTSnDTE bits of the DMA transfer status register (DTSn). The following shows the relationship between the setting of each of the above bits and whether a DMA transfer request is acknowledged.

Table 6-27 Valid DMA transfer request conditions on channel n

Register.bit name	DTSn. DTSnDTE	DTSn. DTSnTC	DTCTn.MLE	DTRCx. DTRCxERR	DTRCx. DTRCxADS	DMA transfer request
When DMA transfers are disabled	0	X	X	X	X	Invalid
When a DMA transfer error occurs	X	X	X	1	X	Invalid
When a DMA transfer is being suspended	X	X	X	X	1	Invalid
When a DMA transfer is completed (Multilink disabled)	X	1	0	X	X	Invalid
When a DMA transfer is completed or not completed (Multilink enable)	1	X	1	0	0	Valid
When DMA transfers are enabled	1	0	0	0	0	Valid

Note n = 0 to 15

6.6.5 Next address function

(1) Next address setting register

This register is used to specify parameters for the transfer to be executed next during a DMA transfer. These parameters are copied to the corresponding register at the beginning of the last DMA cycle. The following registers are provided:

- DMA next source address registers (DNSAnH and DNSAnL)
- DMA next source chip select register (DNSCn)
- DMA next destination address registers (DNDAnH and DNDAnL)
- DMA next destination chip select register (DNDCn)
- DMA next transfer count register (DNTCn)

Each one of these registers has a valid bit in the most significant bit (the most significant bit on the H side in the address register), and whether to copy the transfer parameters to the current register at the beginning of the last DMA cycle can be selected. When the parameters for the transfer to be executed next is copied to the current register, the valid bit is cleared.

(2) Processing upon DMA transfer completion when using next the address function

When a DMA transfer finishes, the DMA transfer enable bit (DTSnDTE) is cleared at the same time the DMA transfer completion status bit (DTSnTC) of the DMA transfer status register (DTSn) is set, and subsequent DMA transfer requests are no longer acknowledged. However, if the multilink enable bit (MLE) is set, DTSnDTE is not cleared and DMA transfer requests can be acknowledged even if DTSnTC is set. Therefore, when using the next address function, if the MLE bit is set in advance, clearing DTSnTC and setting DTSnDTE upon DMA transfer completion can be omitted.

(3) Timing at which the next address is set

The next address setting register can always be rewritten. However, to prevent a conflict between copying to the current register and a write operation by the user, set up the next address setting register before the last DMA cycle starts.

Use of the DMA transfer count match interrupt is recommended as the trigger for setting the next address setting register. In this case, specify the DMA transfer count compare register (DTCCn) so as to secure the time required for setting the next address setting register.

6.6.6 Suspending and resuming a DMA transfer

(1) Suspending or resuming a DMA transfer on all channels by software

By setting the DMA transfer abort bit (DTRCxADS) of the DMA transfer request control register (DTRCx), the next DMA transfer and those that follow can be suspended. During a DMA cycle, the next DMA transfer is suspended after the ongoing DMA cycle has been completed.Note that the DMA transfer enable bit (DTSnDTE) and the software DMA transfer request bit (DTSnSR) of the DMA transfer status register (DTSn) are not cleared.

To resume the suspended DMA transfer, clear the DTRCxADS bit.If a DMA transfer is requested at that point, the transfer of the channel having the highest priority at that time is executed.To end DMA transfer, clear the DMA transfer request with the DTSnDTE bit cleared.

(2) Suspending or resuming DMA transfer by using DMA transfer enable bit (DTE)

By clearing the DMA transfer enable bit (DTSnDTE of the DMA transfer status register (DTSn), the next DMA transfer and those that follow can be suspended. During a DMA cycle, the next DMA transfer is suspended after the ongoing DMA cycle has been completed. Note that the software DMA transfer request bit (SR) of DTS is not cleared.

To resume the suspended DMA transfer, set the bit.If another channel is not executing DMA transfer at that point, the priority is identified as usual.If another channel is executing DMA transfer, the priority is identified after that transfer has been completed. To finish the DMA transfer, clear the DMA transfer request with the DTSnDTE bit cleared.

(3) Suspending or resuming DMA transfer by using software DMA transfer request bit (DTSnSR)

By clearing the software DMA transfer request bit (DTSnSR) of the DMA transfer status register (DTSn), the next DMA transfer and those that follow can be suspended. During a DMA cycle, the next DMA transfer is suspended after the ongoing DMA cycle has been completed.

To resume the suspended DMA transfer, set the DTSnSR bit. If another channel is not executing a DMA transfer at this time, the priority is identified as usual. If another channel is executing a DMA transfer, the priority is identified after the transfer has been completed.

6.6.7 Error responses

(1) Suspending a DMA transfer by using an error response

When an error occurs at the DMA transfer source or transfer destination, the DMA transfer abort bit (DTRCxADS) of the DMA transfer request control register (DTRCx) is set to abort the next and subsequent DMA transfers. At the same time, the DMA transfer error status bit (DTRCxERR) is set and a SysError exception is generated by the CPU. The user can learn on which channel the error has occurred, by using the DMA transfer error flag (DTSnER) of the DMA transfer status register (DTSn), when the user has confirmed that DTRCxERR has been set.

Note that, if an error response is acknowledged in a read cycle, the write cycle is not executed, but the transfer address and the transfer count are updated.

(2) Canceling transfer suspension by using an error response

Suspending a DMA transfer can be canceled by clearing the DMA transfer abort bit (DTRCxADS) and DMA transfer error status bit (DTRCxERR) of the DMA transfer request control register (DTRCx).

Clear the DMA transfer enable bit (DTSnDTE) of the DMA transfer status register (DTSn) in advance, so that the DMA transfer is not resumed after its suspension has been canceled. For a software DMA transfer request, also clear the software DMA transfer request bit (DTSnSR).

6.6.8 Standby mode

When a stop mode request is generated, a DMA transfer is suspended temporarily and the stop mode is entered. Unlike DMA transfer suspension caused by software, this does not affect the DMA control registers. The DMA transfer resumes when the stop request is canceled, and if a DMA request is being held, the DMA transfer starts.

6.7 DTFR Features

The DMA trigger factor register (DTFR) is used to select DMA trigger factors from interrupt signals and to request the DMAC for DMA transfer. The DTFRn registers (n = 15 to 0) are included for selecting the signals to be used for DMA transfer requests from the 128 input interrupt signals.

6.7.1 Features

Number of transfer factors DMA transfer requests (for 16 channels) are selected from 128 interrupt signals.

DMAC interface DMA transfer request signal n is output (n = 15 to 0). DMA transfer request signal n is cleared by an acknowledge signal from the DMAC.

CPU interface The last transfer signal from the DMAC is output as a CPU interrupt signal.

Clearing a transfer request A function that clears transfer request signals sent to the DMAC by way of register access is provided.

Confirming a transfer request A function that clears transfer request signals sent to the DMAC by way of register access is provided.

6.8 DTFR Control Registers

6.8.1 DTFRn - DTFRn register (n = 0 to 15)

Access This register can be read or written in 16-bit units.

Address DTFR0: FFFF7B00$_H$, DTFR1: FFFF7B02$_H$, DTFR2: FFFF7B04$_H$, DTFR3: FFFF7B06$_H$, DTFR4: FFFF7B08$_H$, DTFR5: FFFF7B0A$_H$, DTFR6: FFFF7B0C$_H$, DTFR7: FFFF7B0E$_H$, DTFR8: FFFF7B10$_H$, DTFR9: FFFF7B12$_H$, DTFR10: FFFF7B14$_H$, DTFR11: FFFF7B16$_H$, DTFR12: FFFF7B18$_H$, DTFR13: FFFF7B1A$_H$, DTFR14: FFFF7B1C$_H$, DTFR15: FFFF7B1E$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
REQEN	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	IFCn6	IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-28 DTFRn register contents

Bit position	Bit name	Function
15	REQEN	Enable or disable the DMA source selector on channel n. 1: Enable the source selector 0: Disable thesource selector. Do not issue DMA transfer requests (DMARQ). The settings of IFCn6 to IFCn0 are valid. Requests are always sampled.
6 to 0	IFCn6 to IFCn0	Select the transfer source. For details about the setting value, see *Table 6-2 “DMA start sources (0 to 63)” on page 332* and *Table 6-3 “DMA start sources (64 to 127)” on page 333*.

Caution Stopping DMA channel n by clearing DTFRn.REQEN to 0 does not clear any DMA request held pending on the channel. To also clear a DMA request held pending, set DRQCLR.RQCRn to 1.

6.8.2 DRQCLR - DMA request clear register

Access This register can be read or written in 16-bit units.

Address $FFFF7B40_H$

Initial value 0000_H

15	14	13	12	11	10	9	8
RQCR15	RQCR14	RQCR13	RQCR12	RQCR11	RQCR10	RQCR9	RQCR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RQCR7	RQCR6	RQCR5	RQCR4	RQCR3	RQCR2	RQCR1	RQCR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-29 DRQCLR register contents

Bit position	Bit name	Function
15 to 0	RQCR15 to RQCR0	1: Clear thetransfer request held on channel n.

Note Writing 0 to bits 15 to 0 is ignored.

6.8.3 DRQSTR - DMA request check register

Access This register is read-only, in 16-bit units.

Address $FFFF7B44_H$

Initial value 0000_H

15	14	13	12	11	10	9	8
RQST15	RQST14	RQST13	RQST12	RQST11	RQST10	RQST9	RQST8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
RQST7	RQST6	RQST5	RQST4	RQST3	RQST2	RQST1	RQST0
R	R	R	R	R	R	R	R

Table 6-30 DRQSTR register contents

Bit position	Bit name	Function
15 to 0	RQST15 to RQST0	DMA transfer request status flag 1: Request issued (DMA transfer request signal n is 1.) 0: Request issued (DMA transfer request signal n is 0.)

Chapter 7 Flash Memory

Flash memory type

The V850E2/Sx4-H microcontrollers are equipped with flash memory of the type UX6LF. For details about the UX6LF flash memory, see the Self-Programming and other documents.

The following V850E2/Sx4-H devices are equipped with the following internal flash memory:

Series name	Product name	Code flash	Data flash
SG4-H-1M	μPD70F4013	1 MB	32 KB
SG4-H-1.5M	μPD70F4014	1.5 MB	32 KB
SJ4-H-1M	μPD70F4015	1 MB	32 KB
SJ4-H-1.5M	μPD70F4016	1.5 MB	32 KB
SK4-H-1.5M	μPD70F4017	1.5 MB	32 KB
SK4-H-2M	μPD70F4018	2 MB	32 KB

The code flash memory is connected to the dedicated instruction fetch bus of the V850E2 CPU core. It is used for non-volatile storage of program code and constant data.

The data flash memory is accessible via the memory interface bus. It holds nonvolatile user-created data that is modified during normal program operation.

Flash memory is generally used in the following development environments and applications:

- For modifying software after solder-mounting the microcontroller on the target system
- For differentiating software in small-scale production of various models
- For data adjustment when starting mass production
- For facilitating inventory management
- For updating software after shipment

The flash memory can be written in the following ways:

- By being mounted on the target board by connecting a dedicated flash memory programmer to the target system (serial programming).
- By using the microcontroller's application software (this is known as self-programming).

Additionally, a flash memory address space is provided to store various configuration settings, called flash mask options. Startup settings can be configured for modules such as the clock generator and the watchdog timer by using these options. The flash mask options can be written by using an external flash memory programmer or in self-programming mode. They are not accessible via the normal CPU address space.

7.1 Code Flash Memory Overview

7.1.1 Code flash memory features

- All-block or multiple-block batch erase or single-block erase
- Erase/write by using a single power supply
- Various programming modes:
  - Serial programming by using a dedicated flash memory programmer or by using a flash memory programmer that uses a dedicated serial interface
  - Programming by using the self-programming feature

7.1.2 Code flash memory mapping

The microcontroller's internal code flash memory area is divided into blocks of 4 KB and can be programmed and erased in block units.

The following tables list the block structures and address assignments for all V850E2/Sx4-H devices with code flash memory.

Table 7-1 V850E2/Sx4-H code flash memory configuration

		Block 511 (4 KB)	**001F FFFF$_H$** **001F 0FFF$_H$**
		...	**...**
		Block 384 (4 KB)	**0018 0FFF$_H$** **0018 0000$_H$**
	Block 383 (4 KB)	Block 383 (4 KB)	**0017 FFFF$_H$** **0017 F000$_H$**
	...	...	**...**
	Block 256 (4 KB)	Block 256 (4 KB)	**0010 0FFF$_H$** **0010 0000$_H$**
Block 255 (4 KB)	Block 255 (4 KB)	Block 255 (4 KB)	**000F FFFF$_H$** **000F F000$_H$**
...	...	...	**...**
Block 1 (4 KB)	Block 1 (4 KB)	Block 1 (4 KB)	**0000 1FFF$_H$** **0000 1000$_H$**
Block 0 (4 KB)	Block 0 (4 KB)	Block 0 (4 KB)	**0000 0FFF$_H$** **0000 0000$_H$**
1 MB	1.5 MB	2 MB	**Code flash size**
4/8/16/32/64/128/256 KB			**Boot swap cluster sizes**
• SG4-H: µPD70F4013 • SJ4-H: µPD70F4015	• SG4-H: µPD70F4014 • SJ4-H: µPD70F4016 • SK4-H: • µPD70F4017	• SK4-H: µPD70F4018	**Product**

Address

7.1.3 Data flash memory mapping

The data flash memory is organized in blocks of 2 KB.

The following tables list the block structures and address assignments for all V850E2/Sx4-H devices with data flash memory.

Table 7-2 V850E2/Sx4-H data flash memory

Product	Data flash size	Number of 2 KB blocks	Address range
• SG4-H: - μPD70F4013 - μPD70F4014 • SJ4-H: - μPD70F4015 - μPD70F4016 • SK4-H: - μPD70F4017 - μPD70F4018	32 KB	16	$0200\ 0000_H$ to $0200\ 7FFF_H$

7.2 Code Flash Memory Functional Overview

Serial programming The microcontroller's internal flash memory can be rewritten by using the rewrite function of the dedicated flash memory programmer, while the microcontroller is mounted on the target system or device.

Self-programming Self-programming, in which the flash memory is rewritten by using a user-created program, is ideal for program updates after production or shipment, because no additional programming equipment is required. During self-programming, some types of software processing as well as interrupt servicing can still be used for purposes such as maintaining communication with other devices.

While the self-programming mode can be initiated from the normal operation mode, the external flash memory programming mode can only be entered immediately after a system reset ends.
For how to enter normal operation mode or serial flash programming mode, see *7.4.4 “Flash memory programming control” on page 399*.

Extra area The flash memory contains an extra area, used to store the security and write protection settings, as well as the initial settings for various modules.

Boot swapping Boot swapping allows the flash memory to be re-programmed safely and is used to maintain an operable software version if re-programming fails for any reason, for example if the power fails.
For details about boot swapping, see *7.5.3 “Ensuring safe self-programming (by using boot cluster swapping)” on page 409*.

Protection Various protection flags can be set to prohibit reading, writing, erasing, or otherwise accessing the flash memory during flash memory programming. By using these protection measures, the code flash memory can be protected from being read and re-written by unauthorized persons.
For details about data protection, see *Chapter 11* “*Code Protection and Security*”.

Table 7-3 Flash memory write methods

Environment	Interface	Overview	Operation mode
Serial programming	Dedicated serial interface (FLUR, FLCS)	Flash memory programming is performed by using an external flash memory programmer. The device is mounted on the target system. A dedicated serial interface is used for communication between the device and the flash memory programmer. For details, see *7.4 "Flash Programming by Using Flash Programmer" on page 395*.	Flash memory programming mode
Self-programming	Self-programming library	Flash memory can be rewritten by executing a user-created program that has been written to the flash memory by using serial programming. The self-programming library provides all the functions that need to be called by the application software. For details, see *7.5 "Code Flash Self-Programming" on page 406*.	Self programming mode

Table 7-4 "Basic functions for modifying the flash memory" on page 392 summarizes the functions used to modify the flash memory contents.

Table 7-4 Basic functions for modifying the flash memory

Function	Description	Support (√: Supported, ×: Not supported)	
		Serial programming	Self-programming
Block erasure	The specified memory blocks are erased.	√	√
Chip erasure[a]	The contents of the flash memory area are erased all at once. The extra area (except the boot cluster protection flag) is also erased. **Caution** The chip erase function also erases the data flash memory.	√	x
Write	Writing to the specified addresses and a verify check to see if the write level is secured are performed.	√	√
Erase/write	A 32 KB block is erased while another 32 KB block is written.	√	√
Verify	Data read from the flash memory is compared with data transferred from the flash memory programmer.	√	x[b]
Checksum	The checksum internally calculated by the microcontroller for the entire flash memory is compared with the checksum calculated by the serial programmer	√	x
Blank check	The erasure status of the entire memory is checked.	√	√
Protection settings	The following functions can be prohibited: • Chip erase • Block erase • Write • Read • Rewriting the boot cluster • Specifying a flash shield	√	√ [c]

a) Chip erasure is not possible if boot block protection is enabled (by setting the boot block cluster protection flag) or chip erasure is disabled (by setting the chip erase protection flag).

b) Can be carried out by the user-created program.

c) In self-programming mode, the protection settings, except protection against rewriting of the boot cluster, are disabled.
Rewriting of the boot cluster can be enabled in self-programming mode, but once enabled, it cannot be disabled.

The following table lists the available flash memory protection functions.

For details, see *Chapter 11* "*Code Protection and Security*".

Table 7-5 Protection functions

Function	Description	Applicable (√: Yes, ×: No)	
		Serial programming	Self-programming
Chip erase command prohibition	Erasing the entire flash memory (including the extra area[a] and the data flash) or any block in the flash memory is impossible.	√	×
Block erase command prohibition	Erasing a single block is impossible.	√	×
Program command prohibition	Erasing and rewriting a single block is impossible.	√	×
Read command prohibition	Reading any flash content is impossible.	√	×
Boot area rewriting prohibition	Erasing (by means of block erase or chip erase) or writing the boot cluster is impossible.	√	√
Flash shield	Rewriting and erasing windows other than those specified is impossible.	√	√

a) The boot cluster protection flag is not erased.

7.2.1 Code flash memory erasure and rewrite

Erasure Depending on its block structure, the flash memory can be erased in one of two modes.

- All-block batch erasure (chip erase, available only in serial programming mode)
  All blocks are erased at the same time.
- Block erasure
  Each 4 KB flash memory block can be erased separately.
  In self-programming mode, any number of continuous flash memory blocks can be erased at the same time.

Rewrite In self-programming mode and serial programming mode, it is possible to rewrite the flash memory in units smaller than one block. Once a complete block has been erased, it can be rewritten in 16-byte units. Each unit can be rewritten only once after erasure of the complete block.

Erase/write Erase/write mode allows you to erase a 32 KB block of flash memory while writing to another 32 KB block.

7.3 Data Flash Memory

The V850E2/Sx4-H microcontrollers contain data flash memory in addition to code flash memory.

7.3.1 Data flash memory features

The data flash has the following features:

- Divided into 2 KB blocks
- Can be written in 32-bit steps
- Can be erased in 2 KB blocks
- Can be written or erased while application code from the code flash memory is executing

7.3.2 Data flash writing

The data flash can be written by using the data flash library or by using an external flash memory programmer to execute serial programming.

Programming during normal operation is possible by using the data flash access layer software library. The data flash access layer is described in a separate user's manual.

Note A chip erase command from an external programmer also erases the data flash.

7.4 Flash Programming by Using Flash Programmer

A dedicated flash memory programmer can be used to write to the flash memory externally in serial programming mode.

Serial programming

During serial programming, the microcontroller remains mounted on the board. The board is equipped with a connector via which the flash memory programmer is connected to the target microcontroller.

The microcontroller must be fully functional and operational in serial flash programming mode. Specifically:

- All external power supplies must be active.
- An external resonator must be connected to the X1/X2 pins.

All other necessary microcontroller configurations are executed by the on-chip firmware, and are processed in serial flash programming mode.

Caution Connecting the flash memory programmer to the on-board microcontroller might create conflicts with other signals. For details, see *7.4.4 (2) "Potential conflicts with on-board signal connections" on page 401*.

7.4.1 Programming environment

The recommended environment in which to write data to the flash memory in the microcontroller is shown below.

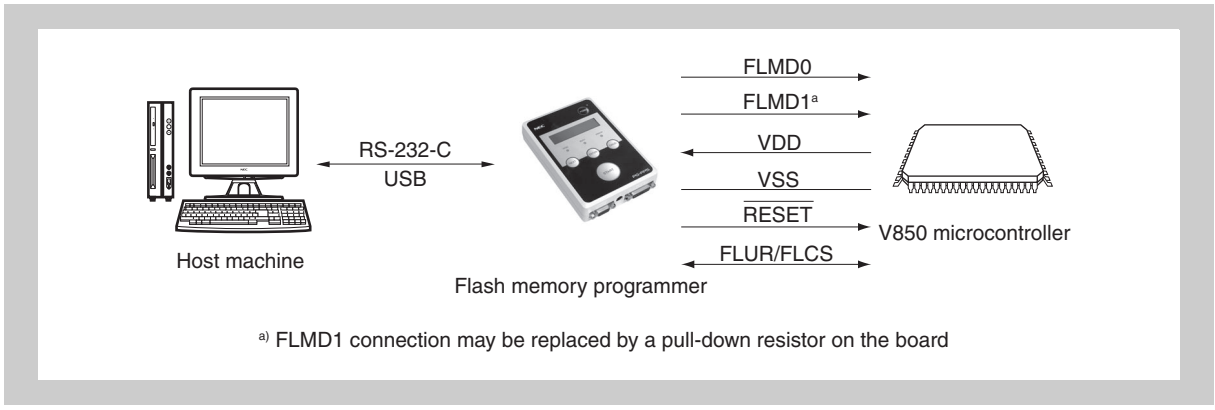


Figure 7-1 Environment in which to write programs to flash memory

A host machine is required for configuring the flash memory programmer. The used flash memory programmer might also feature a stand-alone mode, so a host machine might not be necessary.

The following dedicated microcontroller serial interfaces can be used as the interface between the flash memory programmer and the microcontroller:

- Single-wire asynchronous serial interface FLUR0
- Clocked serial interface FLCS0

Note In normal operation mode, that is, not in flash programming mode, the serial interfaces FLUR0 and FLCS0 cannot be used. The ports used in flash programming mode are specified in subsequent sections.
These are automatically configured for communicating with the flash memory programmer in flash programming mode.

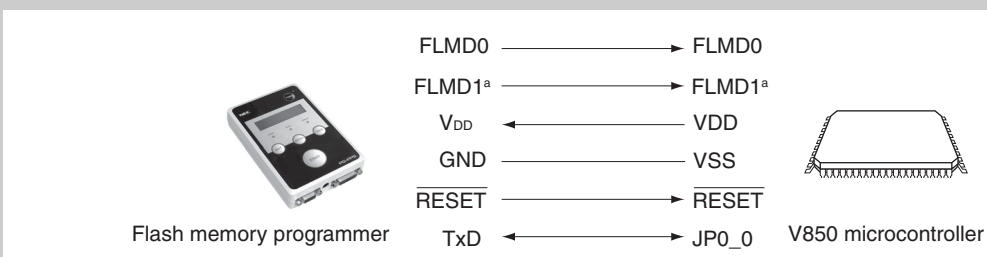
7.4.2 Communication modes

(1) Asynchronous flash programming interface FLUR0

The single-wire asynchronous serial programming interface FLUR0 uses the following port for connecting to the flash memory programmer:

- JP0_0: Data reception/transmission

The external flash memory programmer provides a range of baud rates.



a) FLMD1 connection may be replaced by a pull-down resistor on the board

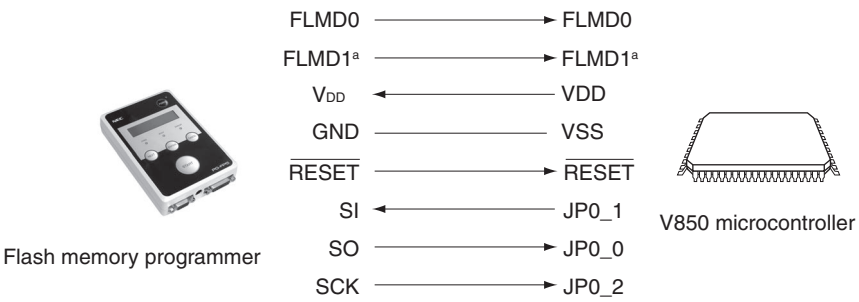
Figure 7-2 Communication with a flash memory programmer by using FLUR0

(2) Synchronous flash programming interface FLCS0

The synchronous serial programming interface FLCS0 uses the following ports for connecting to the flash memory programmer:

- JP0_0: Serial data input
- JP0_1: Serial data output
- JP0_2: Serial data clock input

The external flash memory programmer provides a range of clock rates.



a) FLMD1 connection may be replaced by a pull-down resistor on the board

Figure 7-3 Communication with a flash memory programmer by using FLCS0

The flash memory programmer outputs the serial data clock SCK and the microcontroller functions as a slave.

7.4.3 Pin connections when connecting with flash memory programmer PG-FP5

A connector must be mounted on the target system to connect the flash memory programmer for serial programming. In addition, functions to switch between the normal operation mode and flash memory programming mode and to control the microcontroller's $\overline{\text{RESET}}$ pin must be provided on the board.

When the microcontroller flash memory programming mode is set, all the pins not used for flash memory programming are in their initial states.

When using the PG-FP5 as the flash memory programmer, connect the PG-FP5 target interface connector to the microcontroller as follows:

Table 7-6 Connection of microcontroller and flash memory programmer PG-FP5

Flash memory programmer FG-FP5 pins			Microcontroller signals and ports			
Signal name	I/O	Function	FLUR0		FLCS0	
			Signal	Port	Signal	Port
SO/TxD	O	• FLURS0: Data transmission/ reception • FLCS0: Data transmission	JP0_0		JP0_0	
SI/RxD	I	Reception data	Leave open		JP0_1	
SCK	O	Transfer clock	Leave open		JP0_2	
CLK	O	Clock to microcontroller	Leave open		Leave open	
			Leave open		Leave open	
$\overline{\text{RESET}}$	O	Reset signal	$\overline{\text{RESET}}$		$\overline{\text{RESET}}$	
FLMD0	I	Mode selection	FLMD0		FLMD0	
FLMD1	I	Mode selection	FLMD1[a]/P0_1		FLMD1[a]/P0_1	
H/S	I	Handshake signal	Leave open		Leave open	
V_{DD}	I	Microcontroller supply voltage monitoring	Power supply voltage of JP0 port group buffers[b]		Power supply of JP0 port group buffers[b]	
V_{DD2}	–	Supply voltage	Leave open		Leave open	
V_{PP}	–	Flash programming voltage	Leave open		Leave open	
GND	–	Ground	VSS		VSS	
VDE	–	Reserved	Leave open		Leave open	
RFU-1	–	Reserved	Leave open		Leave open	

a) If FLMD1 is fixed to low level on the target board, the FLMD1 signal can be left unconnected.

b) See *Chapter 38 "Power Supply"* to confirm the correct power supply pins for the JP0 port group in the microcontroller.

For details, see *PG-FP5 User's Manual*, document number R20UT0008EJ*xxxx* (*xxxx* denotes the current version number).

7.4.4 Flash memory programming control

The procedure to program the flash memory is shown below.

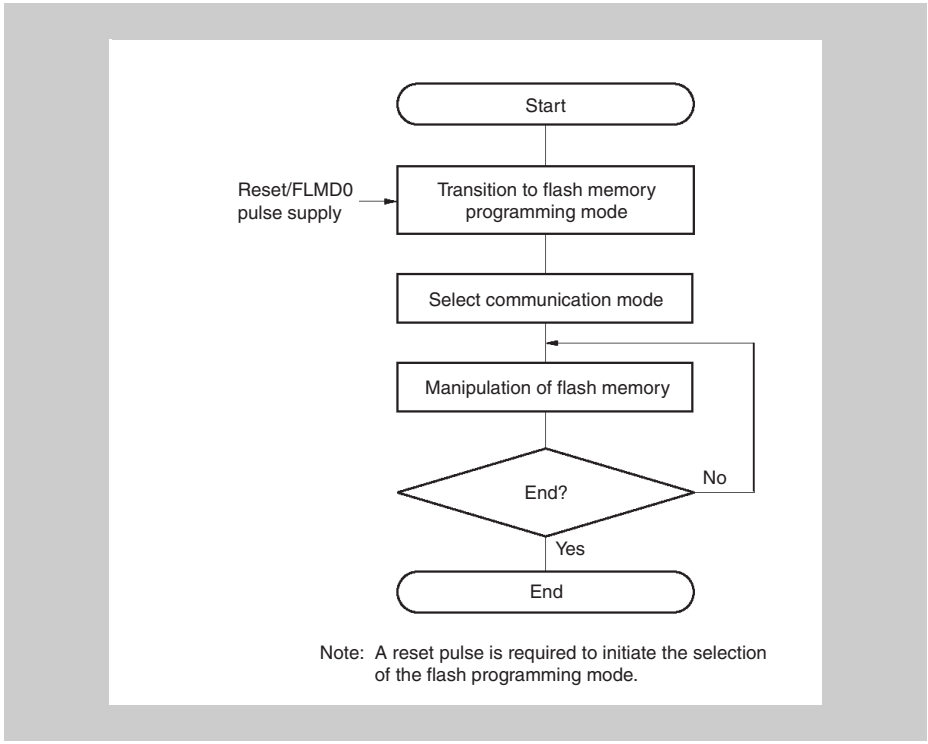


Figure 7-4 Flash memory programming procedure

(1) Operating mode control

To rewrite the flash memory contents by using the flash memory programmer, set the microcontroller to the flash memory programming mode.

To set this mode, specify the FLMD0 and FLMD1 pins as shown in *Table 7-7 "Selection of operating mode" on page 400* and deassert $\overline{\text{RESET}}$.

In the normal operation mode, a low-level voltage is input to the FLMD0 pin. If no flash memory programmer is connected, the pull-down resistor connected to the FLMD0 pin ensures that the system enters normal operation mode.

To enter the serial flash programming mode (that is, on-board programming by using an external flash memory programmer), a high-level voltage must be supplied to the FLMD0 pin and a low-level voltage must be input to the FLMD1 pin after $\overline{\text{RESET}}$ is deasserted.

Table 7-7 Selection of operating mode

Pins		Operating mode
FLMD0	FLMD1 (P0_1)	
L	L	Normal operating mode
H	L	Serial flash programming mode
Other than above		Setting prohibited

Note L: Low-level input

H: High-level input

An example of connecting the FLMD0 and FLMD1 pins is shown below. FLMD1 can be grounded via a resistor. The FLMD1 pin can also be connected directly to the FLMD1 signal of the flash memory programmer.

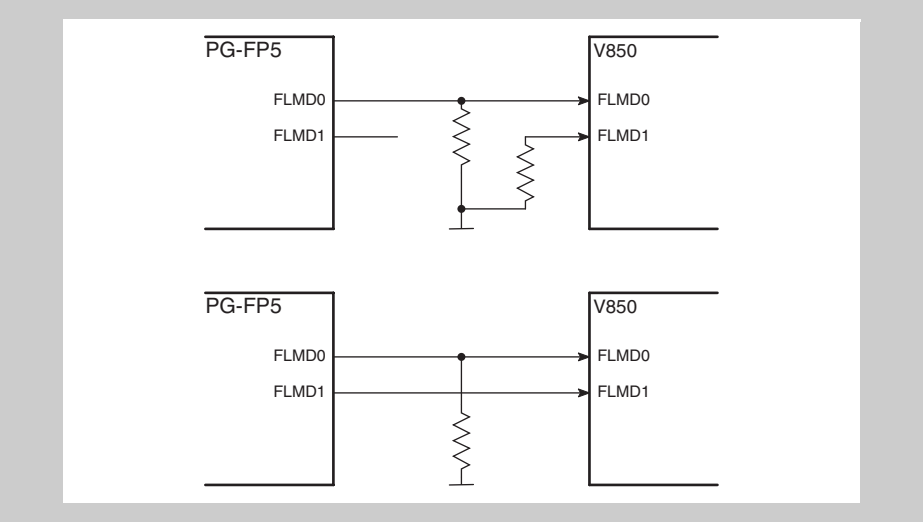


Figure 7-5 Example of connection to flash memory programmer PG-FP5

Once started in normal operating mode (FLMD0 = 0), the FLMD0 pin is used to enable self-programming. Also see *7.5 "Code Flash Self-Programming" on page 406*.

(2) Potential conflicts with on-board signal connections

Serial I/O signals

If other devices are connected to the serial interface pins used for flash memory programming in serial programming mode, make sure that the signals of these devices do not conflict with the signals of the flash memory programmer and the microcontroller. The output pins of the other devices must be isolated or set to a high-impedance state. Ensure that the other devices do not malfunction because of flash memory programmer signals.

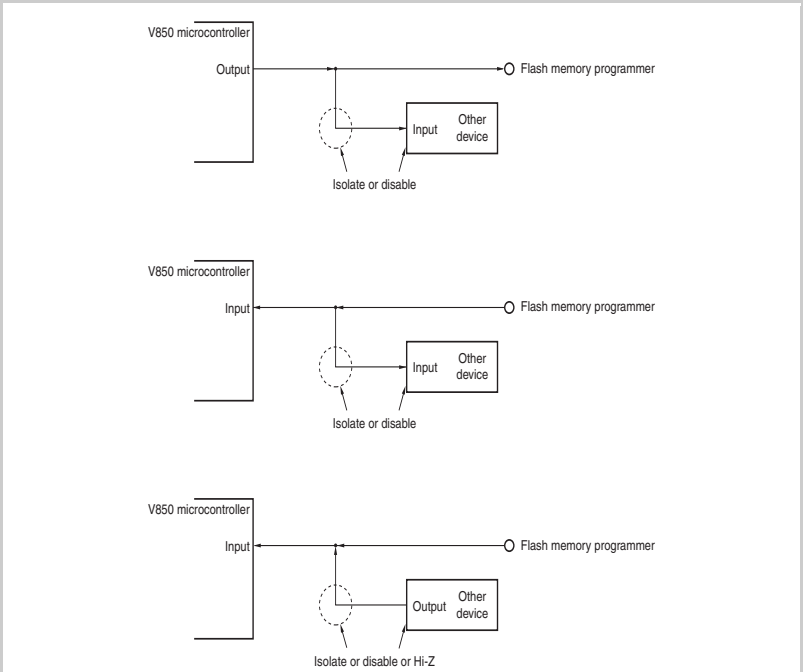


Figure 7-6 Potential conflicts with serial interface signals

$\overline{\text{RESET}}$

Particular care is required if the flash memory programmer's $\overline{\text{RESET}}$ signal is also connected to an on-board reset generator. The reset output from the reset generator might cause flash programming to fail, and therefore might need to be isolated or disabled.

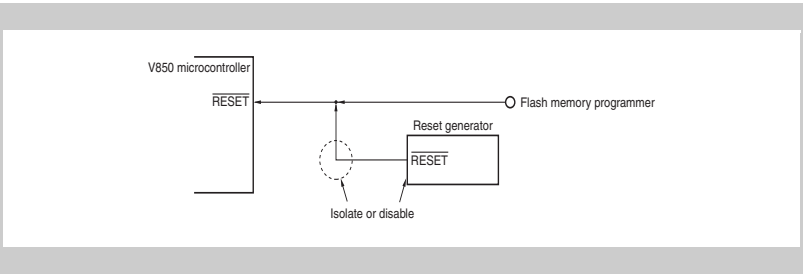


Figure 7-7 Potential conflict with $\overline{\text{RESET}}$

Ports	The V850 port pins are set to the following status during serial programming: The ports used for programming are specified as the FLUR0 and FLCS0 pins. All other pins remain in their default state after a reset ends. If the default state of the pins not used for programming is input port or high-impedance output port, pay attention to other devices connected to these pins. If these devices require specific pin levels, the ports might have to be connected to VDD or VSS via resistors.
Oscillators	Connect all oscillator pins in the same way as in the normal operation mode.
$\overline{\text{DCUTRST}}$	During flash memory programming, input a low level to $\overline{\text{DCUTRST}}$ (port JP0_4) or leave it open. Do not input a high level.
Power supply	Supply the same power to all power supply pins, including reference voltages and power regulator pins, as in the normal operation mode.

(3) Selecting the communication mode

The communication interface is chosen by applying a specified number of pulses to the FLMD0 pin after a reset ends. Note that this is handled by the flash memory programmer.

Figure 7-8 “Selecting the communication mode” on page 403 gives an example of how FLCS0 is used for communication between the flash memory programmer and the microcontroller.

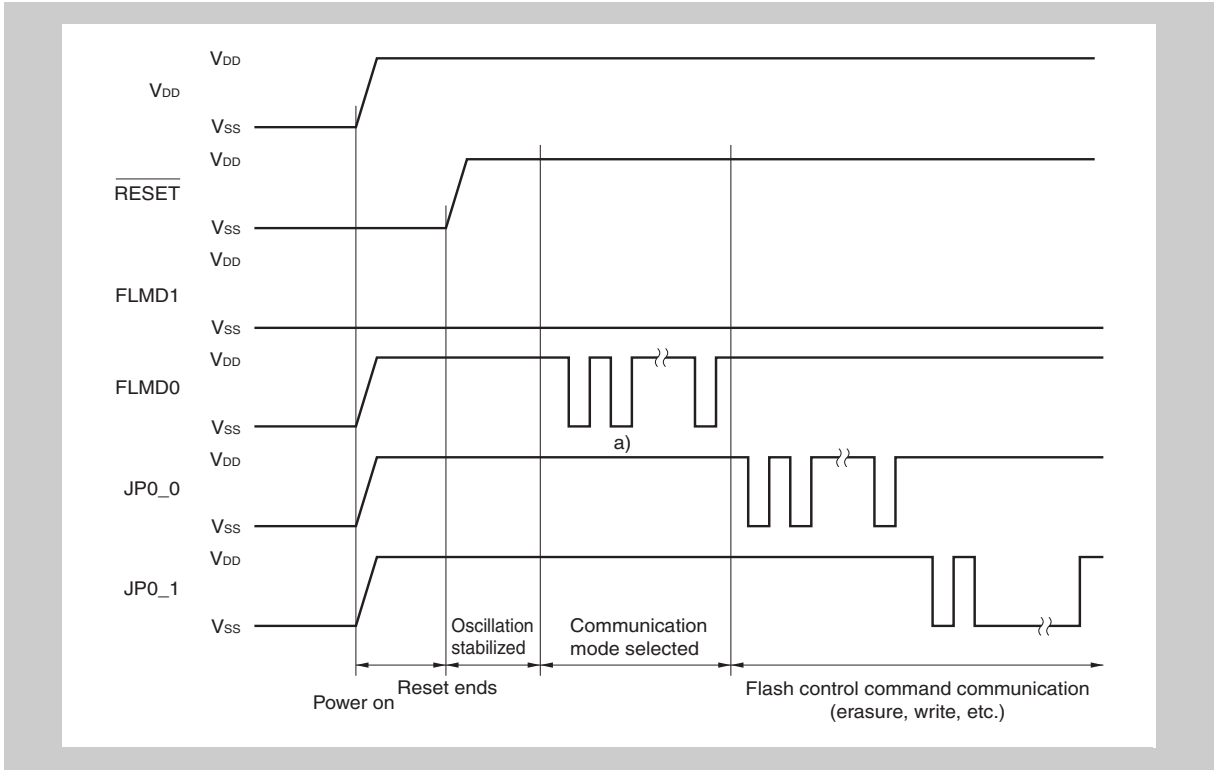


Figure 7-8 Selecting the communication mode

a) The number of clock cycles to be input depends on the selected communication mode. For details, see *Table 7-8 “FLMD0 pulses for specifying the communication mode” on page 403*.

Table 7-8 FLMD0 pulses for specifying the communication mode

FLMD0 pulse	Communication mode	Remarks
0	FLUR0	Communication rate: 9,600 bps (after reset), LSB first
8	FLCS0	Microcontroller operates as a slave, MSB first
Other	–	Setting prohibited

If FLUR0 is selected after the specified number of FLMD0 pulses have been received at 9,600 bps, the flash memory programmer changes the baud rate to the one specified by the user by using the flash memory programmer’s user interface.

(4) Communication commands

The flash memory programmer sends commands to the microcontroller. According to the command, the microcontroller returns status information or the requested data.

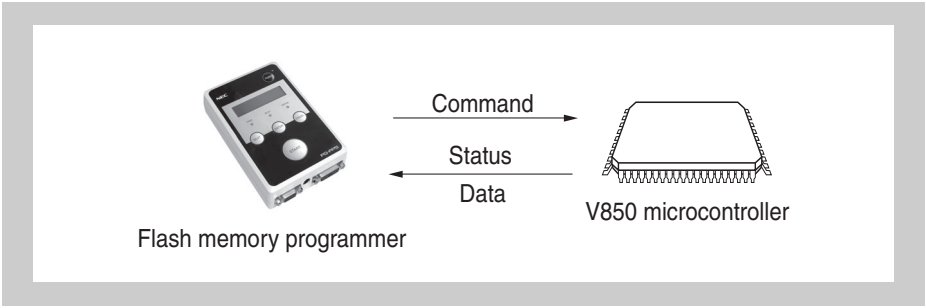


Figure 7-9 Communication command exchange

The following table lists the flash memory control commands of the microcontroller. All these commands are issued by the flash memory programmer, and the microcontroller performs the corresponding processing.

Table 7-9 Flash memory control commands (1/2)

Classification	Command name	Support		Function
		FLCS0	FLUR0	
Blank check	Block blank check	√	√	Checks the erasure status of the entire memory.
Erase	Chip erase	√	√	Erases the contents of all the flash memory areas, including the code flash, data flash, and extra area.
	Block erase	√	√	Erases the memory contents of the specified blocks. Note that the extra area is not erased.
Write	Write	√	√	Writes data based on the write address and the number of written bytes specified by the user, and executes a verify check.
Erase/write	Erase and write	√	√	Erases the specified number of flash blocks in the code flash or data flash. Note that the extra area is not erased.
	Chip erase and write	√	√	Erases and writes all the flash memory areas, including the code flash, data flash and extra area.
Read	Read	√	√	Reads data based on the write address and the number of read bytes specified by the user.
Verify	Verify	√	√	Compares the input data with the memory data.
ID	Set ID code	√	√	Used to specify the on-chip debug ID to the OCDIDL, OCDIDM, and OCDIDH registers.
	Get ID code	√	√	Reads the on-chip debug ID from the OCDIDL, OCDIDM, and OCDIDH registers.

Table 7-9 Flash memory control commands (2/2)

Classification	Command name	Support		Function
		FLCS0	FLUR0	
CRC check	CRC check	√	√	Calculates the checksum for a specified number of flash blocks in the code flash or the data flash. Note that this checksum calculation does not inlcude the extra area.
	Chip CRC check	√	√	Calculates the checksum for all the flash memory areas, including the code flash, data flash and extra area.
Flash mask options	Set flash mask option	√	√	Specifies the flash mask options by using the OPBT0 register.
	Get flash mask option	√	√	Reads the flash mask options from the OPBT0 register.
Protection	Protection setting	√	√	Specifies protection against chip erasure, block erasure, and writing.
	Get protection settings	√	√	Reads the protection settings.
System setting and control	Reset	√	√	Resets all statuses.
	Oscillation frequency setting	√	√	Specifies the oscillation frequency.
	Baud rate setting	–	√	Specifies the baud rate when UART is used.
	Silicon signature	√	√	Reads the silicon signature information.
	Version acquisition	√	√	Reads the version information of the device.

7.5 Code Flash Self-Programming

This V850 microcontroller supports a flash macro service that allows the internal flash memory to be rewritten by a user-created program.

By using this flash macro service and a self-programming library (FSL) provided by Renesas, the user-created program can be used to rewrite the flash memory with data that has been transferred to the internal RAM or the external memory.

Thus the user-created program can be upgraded and constant data fields can be rewritten.

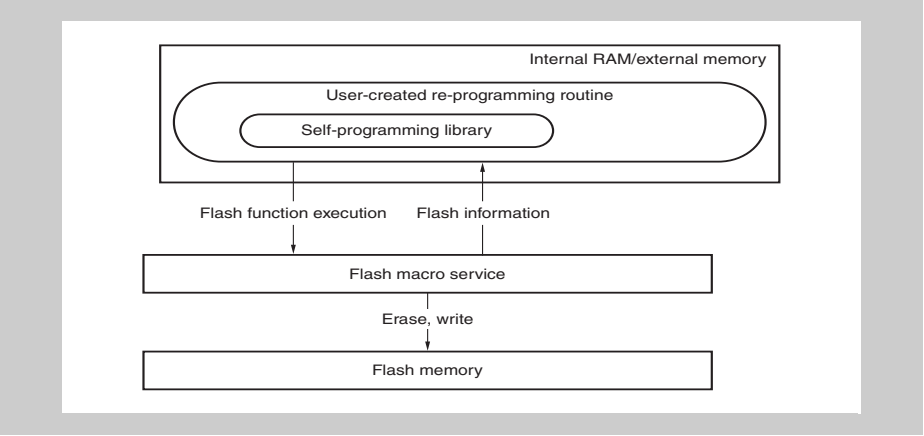


Figure 7-10 Concept of self-programming

The flash memory cannot be accessed during self-programming. Thus program execution is only possible by fetching instructions from the internal RAM or external memory.

Consequently, the instructions of user-created re-programming software routines that will remain running during self programming must be copied from the flash memory to the internal RAM or external memory prior to executing self-programming. Because interrupt servicing using the interrupt vectors in the flash memory is also impossible during self-programming, a special feature is provided to re-route interrupt acknowledgments to the internal RAM (see *7.5.4 “Interrupt servicing during flash self-programming” on page 413*).

See the Self-Programming application note for a comprehensive description of flash self-programming. This application note also explains the features of the self-programming library. Note that the V850E2/Sx4-H microcontrollers are equipped with UX6LF flash memory.

7.5.1 Enabling self-programming

Self programming can be initiated from the normal mode of the microcontroller.

Self-programming must be enabled to avoid problems such as unintended flash re-programming. The following two methods are provided to enable self-programming:

- Setting the external FLMD0 pin to high level
  This requires some external components or wiring, such as connecting an output port to FLMD0.
- Setting the internal register bit FLMDCNT.FLMDPUP.
  This method does not need any special external components or wiring.

The following register is used to enable self-programming internally by using software:

(1) FLMDCNT – FLMD control register

This register is used to control the internal pull-up and pull-down resistors connected to the FLMD0 pin, thus enabling or disabling self-programming.

Protection This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register FLMDPCMD.
For how to write to write-protected registers, see *3.10 "Write-Protected Registers"*.

Access This register can be read or written in 8-bit units.

Address FF43 8000$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FLMDPUP
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 7-10 FLMDCNT register contents

Bit position	Bit name	Function
0	FLMDPUP	FLMD0 pull-up/pull-down control 0: Activate pull-down resistor connected to FLMD0 (self-programming mode disabled) 1: Activate pull-up resistor connected to FLMD0 (self-programming mode enabled)

7.5.2 Self-programming library features

The self-programming library is used to help execute self-programming of code flash memory by using a user-created program.

This library provides a set of C function calls used to carry out the following basic processing:

- Flash memory blank check, erase, rewrite, and verify
- Boot cluster swapping (includes boot cluster definition)
- Setting protection flags
- Obtaining various information about the code flash memory

For how to use the library functions, see the Self-Programming application note. Note that the V850E2/Sx4-H microcontrollers are equipped with UX6LF flash memory.

7.5.3 Ensuring safe self-programming (by using boot cluster swapping)

The V850 flash microcontrollers support a mechanism to swap a cluster of code flash memory blocks, starting from address 0000 0000_H, with another cluster of the same size, located immediately above the first one.

Boot swap cluster

The cluster of blocks starting at address 0000 0000_H is called the active boot swap cluster, because it contains the entry point of the user-created program at the default reset vector 0000 0000_H.

Boot swap flag

Which of the two clusters is the active boot cluster is controlled by the boot swap flag, which can be defined by using a self-programming library function during flash programming.
The boot swap flag is stored in the flash memory extra area.

Figure 7-11 "Boot cluster swapping" on page 409 shows an example of swapping the boot cluster with a cluster of four flash memory blocks. After inverting the boot flag, it becomes not(boot_flag), and blocks 4 to 7 become the active boot cluster. Thus the user-created program starts from the new boot swap cluster after the next reset ends.

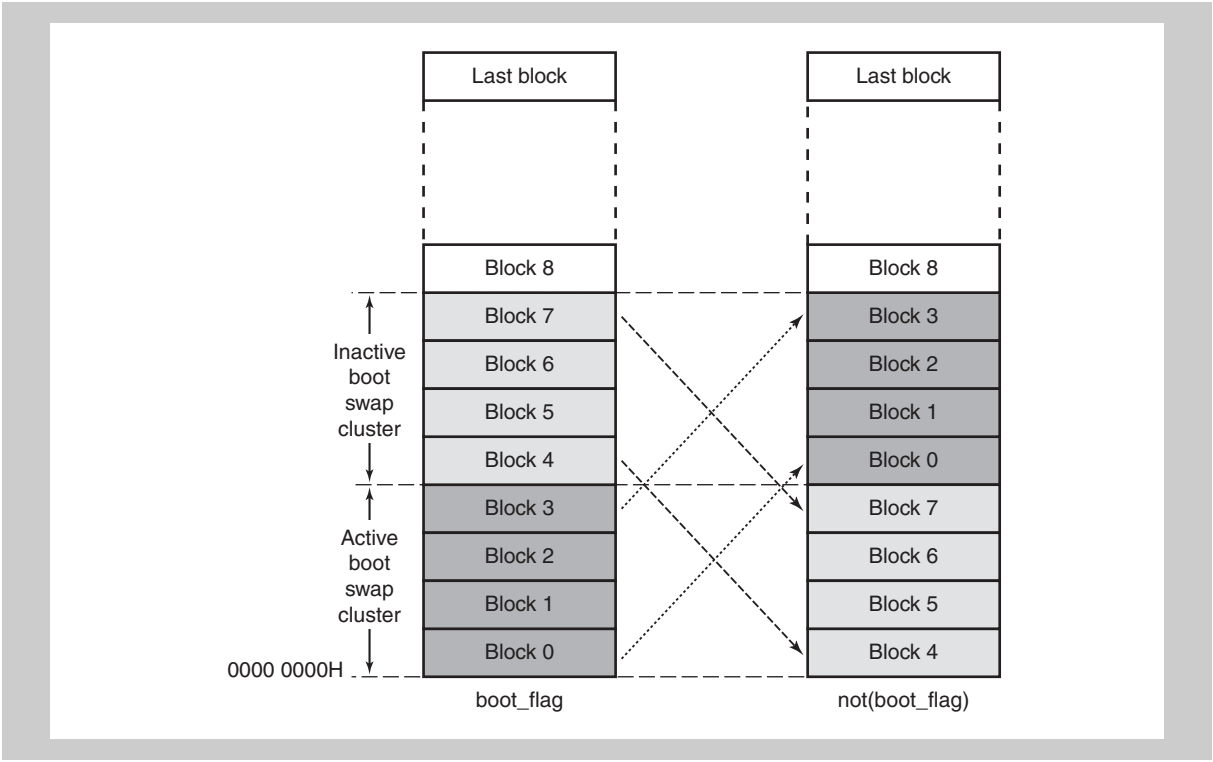


Figure 7-11 Boot cluster swapping

Safe self-programming

Boot cluster swapping assures safe self-programming. If boot code is rewritten, the new code is written to the inactive boot cluster, while the boot flag maintains its previous state.
If rewriting the boot cluster is completed successfully, the boot flag is inverted, making the new boot code active.
If rewriting the new boot code fails due to causes such as power failure or an unintended reset, the old boot code remains active and rewriting can be started again.

Boot cluster

The boot code size might be smaller than the boot swap cluster size.

The flash memory blocks that include the boot code are called the boot cluster. The number of boot blocks in the cluster can be defined by using the self-programming library during self-programming.

The boot swap cluster size is determined by the boot cluster size. This is automatically evaluated based on the number of boot blocks defined during self-programming.

Table 7-11 “Relationship between boot blocks and boot swap cluster” on page 411 shows the relationship between the number of boot blocks, the boot cluster size, and the boot swap cluster.

Number of boot blocks

The number of boot blocks has to be defined by the user during self-programming. This value determines the blocks that are subject to boot cluster protection, which allows the boot blocks to be protected from erasure or writing.

Boot block protection

To prohibit rewriting the boot blocks, set the boot cluster protection flag during flash memory programming. If this flag is set, the blocks of the active boot cluster can neither be erased nor written. Boot cluster swapping is impossible as well.
Note that only the blocks of the active boot cluster are protected. In the example shown in *Figure 7-12 “Boot cluster swapping” on page 412*, erasing and writing blocks 0 and 1 is prohibited, while erasing and writing blocks 2 and 3 is permitted.

Caution Once boot cluster protection is enabled, it cannot be disabled.

For details about the flash memory protection flags, see *Chapter 11* “*Code Protection and Security*”.

Table 7-11 Relationship between boot blocks and boot swap cluster

	Number of boot blocks	Boot cluster size	Boot swapping	Boot cluster protection	
			Active boot swap cluster ↔ Inactive boot swap cluster	Size	Address
	00_H	4 KB	$0000\ 0000_H$ to $0000\ 0FFF_H$ ↔ $0000\ 1000_H$ to $0000\ 1FFF_H$	4 KB	$0000\ 0000_H$ to $0000\ 0FFF_H$
	01_H	8 KB	$0000\ 0000_H$ to $0000\ 1FFF_H$ ↔ $0000\ 2000_H$ to $0000\ 3FFF_H$	8 KB	$0000\ 000_H$ to $0000\ 1FFF_H$
	02_H	16 KB	$0000\ 0000_H$ to $0000\ 3FFF_H$ ↔ $0000\ 4000_H$ to $0000\ 7FFF_H$	12 KB	$0000\ 0000_H$ to $0000\ 2FFF_H$
	03_H			16 KB	$0000\ 0000_H$ to $0000\ 3FFF_H$
	04_H	32 KB	$0000\ 0000_H$ to $0000\ 7FFF_H$ ↔ $0000\ 8000_H$ to $0000\ FFFF_H$	20 KB	$0000\ 0000_H$ to $0000\ 4FFF_H$
	...			...	...
	07_H			32 KB	$0000\ 0000_H$ to $0000\ 7FFF_H$
	08_H	64 KB	$0000\ 0000_H$ to $0000\ FFFF_H$ ↔ $0001\ 0000_H$ to $0001\ FFFF_H$	36 KB	$0000\ 0000_H$ to $0000\ 8FFF_H$
	...			...	...
	$0F_H$			64 KB	$0000\ 0000_H$ to $0000\ FFFF_H$
	10_H	128 KB	$0000\ 0000_H$ to $0001\ FFFF_H$ ↔ $0002\ 0000_H$ to $0003\ FFFF_H$	68 KB	$0000\ 0000_H$ to $0001\ 0FFF_H$
	...			...	...
	$1F_H$			128 KB	$0000\ 0000_H$ to $0001\ FFFF_H$
	20_H	256 KB	$0000\ 0000_H$ to $0003\ FFFF_H$ ↔ $0004\ 0000_H$ to $0007\ FFFF_H$	132 KB	$0000\ 0000_H$ to $0002\ 0FFF_H$
	...			...	...
<R>	$3F_H$			512 KB	$0000\ 0000_H$ to $0007\ FFFF_H$
<R>	40_H	Setting prohibited			
	...				
<R>	FF_H				

Maximum boot swap cluster size

The maximum boot cluster size is 256 KB. Thus code flash exceeding 512 KB is not subject to boot cluster swapping.

Figure 7-12 "Boot cluster swapping" on page 412 shows an example of the boot cluster swap settings:

- The number of boot blocks is 2 (the boot cluster contains 2 blocks), thus the active boot cluster comprises:
  - Blocks 0 and 1 if the boot flag is not inverted (boot_flag).
  - Blocks 4 and 5 if the boot flag is inverted (not(boot_flag)).
- The active boot swap clusters comprises:
  - Blocks 0 to 3 if the boot flag is not inverted (boot_flag).
  - Blocks 4 to 7 if the boot flag is inverted (not(boot_flag)).

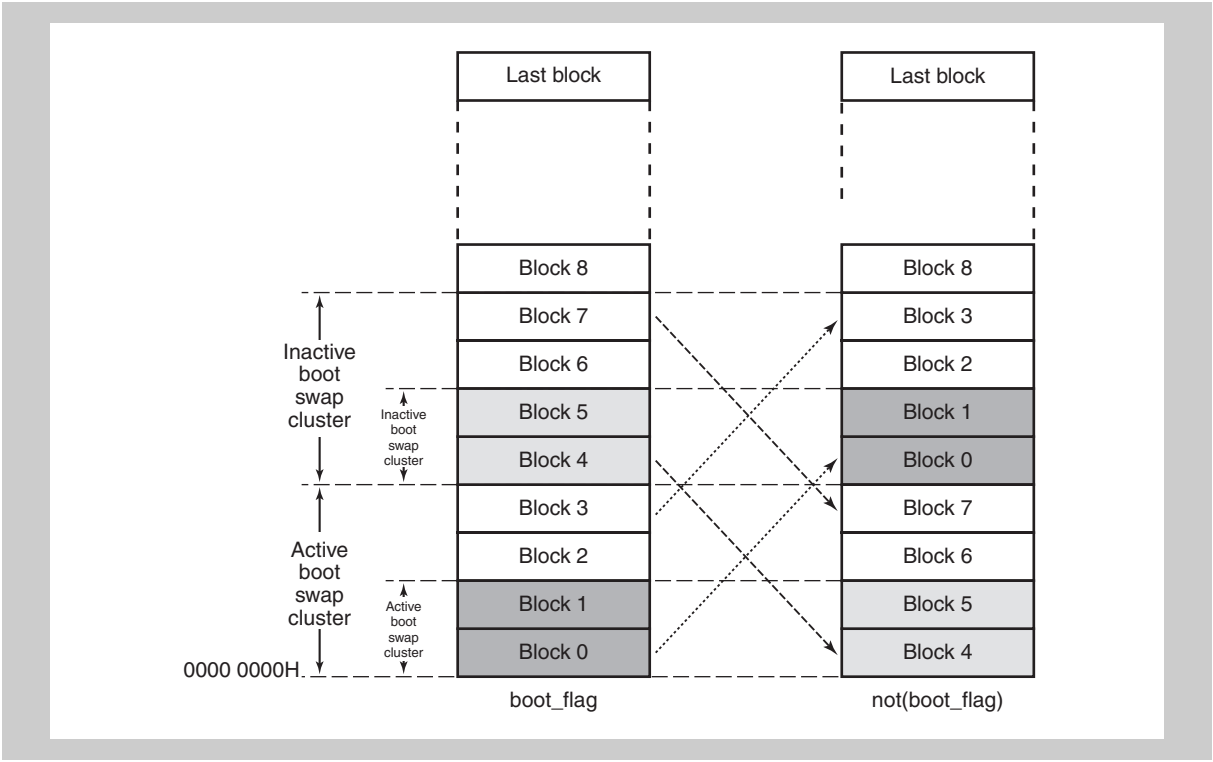


Figure 7-12 Boot cluster swapping

Boot block protection

To prohibit rewriting the boot blocks, set the boot cluster protection flag during flash memory programming. If this flag is set, the blocks of the active boot cluster can neither be erased nor written. Boot cluster swapping is impossible as well.
Note that only the blocks of the active boot cluster are protected. In the example shown in *Figure 7-12 "Boot cluster swapping" on page 412*, erasing and writing blocks 0 and 1 is prohibited, while erasing and writing blocks 2 and 3 is permitted.

Caution Once boot cluster protection is enabled, it cannot be disabled.

For details about the flash memory protection flags, see *Chapter 11* "*Code Protection and Security*".

7.5.4 Interrupt servicing during flash self-programming

This microcontroller provides features to maintain interrupt servicing during self-programming.

Because neither the interrupt vector table nor the interrupt handler routines, which are normally located in the flash memory, are accessible during self-programming, interrupt acknowledgments have to be re-routed to an area other than the flash memory, such as the internal RAM.

Therefore, the following two prerequisites must be satisfied to enable interrupt servicing during self-programming:

- The relevant interrupt handler routine must be copied to an area other than the flash memory, such as the internal RAM. The user must initiate this copy process.
- The relevant interrupt acknowledgment must be re-routed to that handler. Re-routing to the handler is performed by using the CPU registers SW_CFG/SW_BASE and EH_CFG/EH_BASE. For details about data protection, see *V850E2M Architecture User's Manual*.

The interrupt vectors can be re-routed in the following two ways:

- All interrupts can be mapped to a single interrupt vector on interrupt channel 0.
- The base address of the interrupt vector table can be mapped to a different address. In this case, the offsets of the interrupt channels are added to the new base address and the correct interrupt vector is obtained when an interrupt is acknowledged.

For details about flash self-programming, see the Self-Programming application note.

7.6 Flash Mask Options

The flash memory contains an extra area, called the flash mask option area, that holds user-specified data for various purposes.

The flash mask options become effective when the external $\overline{\text{RESET}}$ signal is deasserted, or the power-on-clear reset signal POCRES is asserted and thus the modules are initialized.

Caution If the flash memory is programmed during a debug session by using the on-chip debugger and if any flash mask options are changed, a target reset command must be issued to apply the new option settings.

All flash mask options can be read in all operation modes.

Changing the flash mask options depends partly on the operating mode; some cannot be changed at all.
The following table shows the flash mask options and whether the options can be changed, according to the operating mode.

Table 7-12 Flash mask options and setting (1/2)

Function	Flash mask options	Can be changed?			
		Normal mode	Serial flash programming mode	Flash self-programming mode	Debugging mode
Controlling JTAG port group JP0	OPBT0.OPBT0[31]	No	Yes	Yes	Yes
Controlling connection with MINICUBE2	OPBT0.OPBT0[30:28]	No	Yes	Yes	Yes
Selecting operating mode of MEMC	OPBT0.OPBT0[27]	No	Yes	Yes	Yes
Enabling or disabling VAC of WDTA1	OPBT0.OPBT0[26]	No	Yes	Yes	Yes
Selecting whether to start WDTA1 automatically or by using software	OPBT0.OPBT0[24]	No	Yes	Yes	Yes
Selecting whether to initially enable or disable WDTA1	OPBT0.OPBT0[23]	No	Yes	Yes	Yes
Enabling or disabling VAC of WDTA0	OPBT0.OPBT0[22]	No	Yes	Yes	Yes
Selecting whether to start WDTA0 automatically or by using software	OPBT0.OPBT0[20]	No	Yes	Yes	Yes
Selecting whether to initially enable or disable WDTA0	OPBT0.OPBT0[19]	No	Yes	Yes	Yes

Table 7-12 Flash mask options and setting (2/2)

Function	Flash mask options	Can be changed?			
		Normal mode	Serial flash programming mode	Flash self-programming mode	Debugging mode
Specifying initial value of WDTAn count clock	OPBT0.OPBT0[18:16]	No	Yes	Yes	Yes
Specifying power sequencer timer value	OPBT0.OPBT0[15:3]	No	Yes	Yes	Yes
Selecting bus clock for MEMC	OPBT0.OPBT0[2:1]	No	Yes	Yes	Yes

7.6.1 OPBT0 - Flash mask option register 0

Access In normal operating mode, this register is read-only, in 32-bit units. Writing to this register is only possible in flash programming and self-programming mode.

Address FF47 000C$_H$

Initial value User-defined value

31	30	...		0
OPBT0 [31]	OPBT0 [30]	...	OPBT0[1]	0
R	R	...	R	R

Table 7-13 OPBT0 register contents (1/2)

Bit position	Bit name	Connected to:		Function
		Module	Signal	
31	OPBT0[31]	JTAG port group JP0	OPJTAG	This bit controls the function of JTAG port group JP0. 0: JP0 is used as a general purpose or alternate-function port. 1: JP0 is used as a JTAG port.
30 to 28	OPBT0[30:28]	OCD	MINI2_[2:0]	These bits control the connection with MINICUBE2. 0b100:Enable connection with MINICUBE2. 0b000: Do not use MINICUBE2.
27	OPBT0[27]	MEMC	MEMCMD	This bit selects the operating mode of the MEMC. 0: Multiplexed bus mode 1: Separate bus mode
26	OPBT0[26]	WDTA1	OPWDVAC1	This bit specifies whether to enable or disable the variable activation code function (VAC) of WDTA1. 0: Enable VAC. 1: Disable VAC.
25	OPBT0[25]	Reserved	Reserved	–
24	OPBT0[24]	WDTA1	OPWDRUN1	This bit specifies the setting of OPWDRUN1 (WDTA1 start trigger). 0: Software trigger 1: Automatic start
23	OPBT0[23]	WDTA1	OPWDEN1	This bit specifies the setting of OPWDEN1 (stop or operate WDTA1). 0: Stop 1: Operate
22	OPBT0[22]	WDTA0	OPWDVAC0	This bit specifies whether to enable or disable the variable activation code function (VAC) of WDTA0. 0: Enable VAC. 1: Disable VAC.
21	OPBT0[21]	Reserved	Reserved	–
20	OPBT0[20]	WDTA0	OPWDRUN0	This bit specifies the setting of OPWDRUN0 (WDTA0 start trigger). 0: Software trigger 1: Automatic start

Table 7-13 OPBT0 register contents (2/2)

Bit position	Bit name	Connected to:		Function
		Module	Signal	
19	OPBT0[19]	WDTA0	OPWDEN0	OPWDEN0 (stop or operate WDTA0) 0: Stop 1: Operate
18 to 16	OPBT0[18:16]	WDTA0 WDTA1	OPWDOVF [2:0]	These bits specify the initial value of the WDTAnMD.WDTAnOVF[2:0] bits that select the count clock for WDTA0 and WDTA1.
15 to 3	OPBT0[15:3]	PWRSEQ	TDON[12:0]	These bits specify the power sequencer timer value.
2, 1	OPBT0[2:1]	MEMC	MEMCCK [1:0]	These bits select the bus clock of the MEMC. 01:CKSCLK_000 clock divided by 2[a] 10:CKSCLK_000 clock divided by 4
0	OPBT0[0]	Reserved	Reserved	–

a) This setting is available only when CKSCLK_000 is 80 MHz or below.

Chapter 8 Data CRC Function A (DCRA)

This chapter describes data CRC function A (DCRA).

The first section describes the properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names.
The subsequent sections describe the features that apply to all implementations.

8.1 V850E2/Sx4-H DCRA Features

Instances

This microcontroller has the following number of instances of DCRA:

Table 8-1 Instances of DCRA

DCRA	
Number of instances	1
Name	DCRA0

Instances index n

Throughout this chapter, the individual instances of DCRA are identified by the index “n” (n = 0), for example, DCRAnCTL for the CRCn control register.

Register addresses

All DCRAn register addresses are given as addresses offset from the base address <DCRAn_base>.
The base address <DCRAn_base> of each DCRAn is listed in the following table:

Table 8-2 Register base addresses <DCRAn_base>

DCRAn	<DCRAn_base> address
DCRA0	FF81 F000$_H$

Clock supply

The following clock is supplied to DCRA:

Table 8-3 DCRAn clock supply

DCRAn	Clock	Connected to:
DCRA0	PCLK	Clock generator CKSCLK_101

DCRA hardware reset

DCRA and its registers are initialized by the following reset signal:

Table 8-4 DCRAn reset signal

DCRAn	Reset signal
DCRAn	System reset SYSRES

8.2 Functional Overview

Features summary

Data CRC function A can be used to verify or generate CRC protected data streams of arbitrary length and different bit widths.

- 32-bit Ethernet CRC ($04C11DB7_H$)

  $(X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X^1+1)$
- 16-bit CCITT CRC (1021_H)
- $(X^{16}+X^{12}+X^5+1)$
- CRC generation to an arbitrary data block length
- After initialization of the CRC input register every write access to the CRC input register generates a new CRC according to the chosen polynomial and the result is stored in the CRC data register.

The following picture shows the block diagram of data CRC function A.

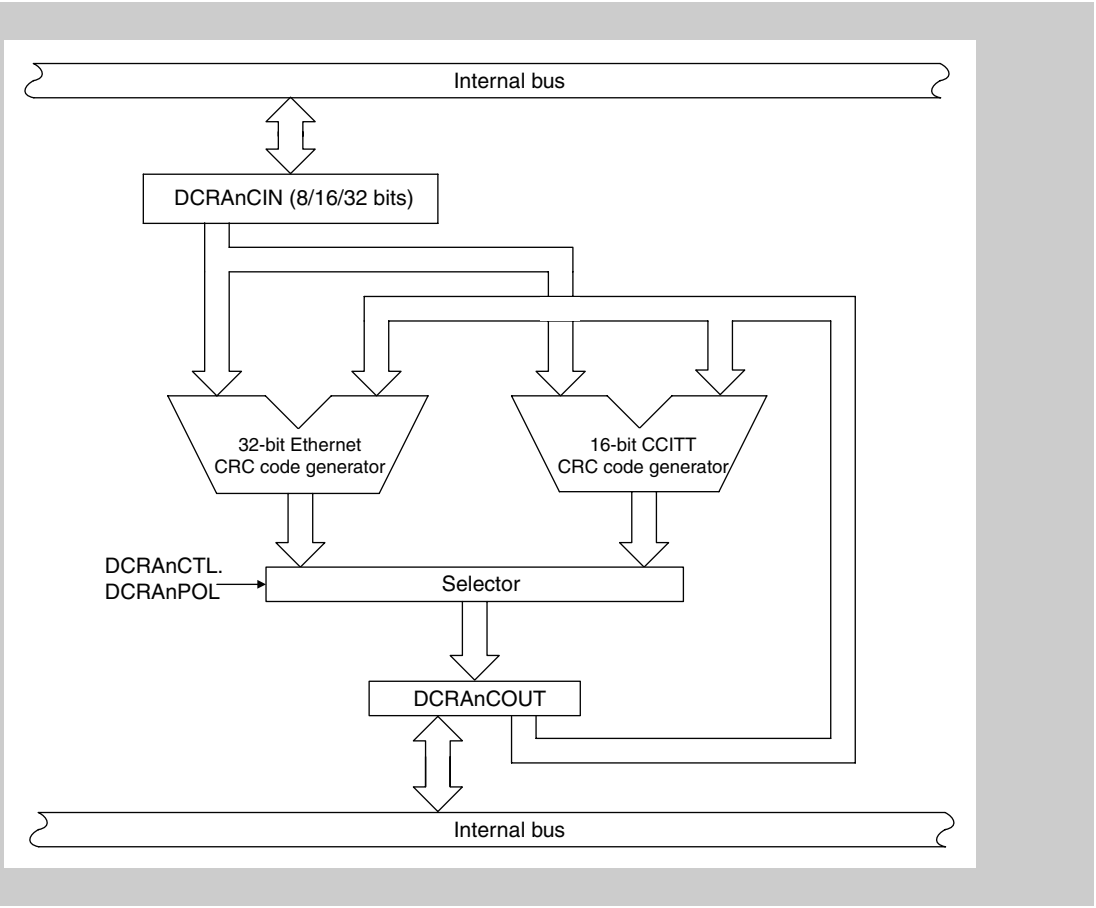


Figure 8-1 Block diagram of Data CRC Function A

8.3 Functional Description

Data CRC function A generates a CRC (cyclic redundancy check) of an arbitrary data block length. The data is forwarded to data CRC function in 8-, 16- or 32-bit units. The CRC polynomial can either be selected for 32-bit Ethernet or 16-bit CCITT, the initial value must be set at the DCRAnCOUT register before the first write access to the CRC input register (DCRAnCIN) is performed.

The flow chart below shows the CRC generating procedure.

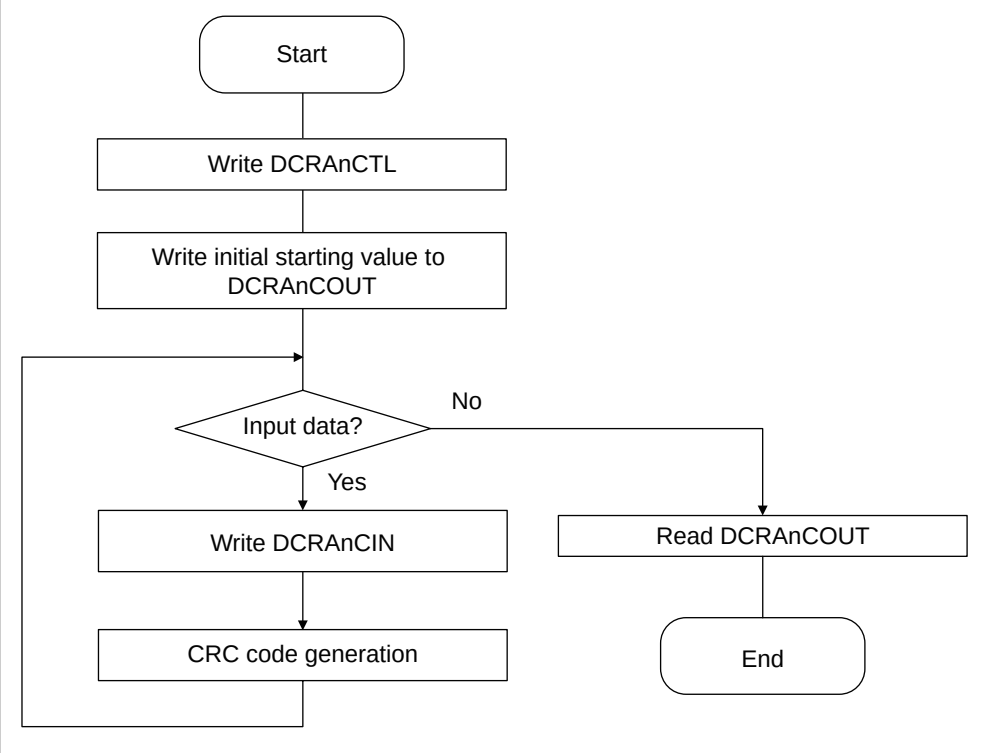


Figure 8-2 Data CRC Function A flow diagram

Note For the setup method and cautions on each register, see *8.4.2 "DCRA register details"*.

8.4 Registers

This section provides a description of all registers of the DCRA.

8.4.1 DCRA register overview

The DCRA is controlled by the following registers:

Table 8-5 DCRA registers overview

Register name	Symbol	Address
DATA-DCRA input register	DCRAnCIN	<DCRAn_base>
DATA-DCRA data register	DCRAnCOUT	<DCRAn_base> + 4_H
DCRA control register	DCRAnCTL	<DCRAn_base> + 20_H

<DCRAn_base> The base address <DCRAn_base> of data CRC function A is defined in the first section of this chapter under the key word “Register addresses”.

8.4.2 DCRA register details

(1) DCRAnCIN - CRC input register

This register holds the input data for the CRC calculation. The effective bit width used for CRC calculation must be set by DCRAnCTL.DCRAnISZ[1:0].

When data is written to this register, the CRC code is generated.

The CRC calculation is immediately started after the DCRAnCIN register is written. The initial starting value must be written to DCRAnCOUT before the first data in the data block is written to DCRAnCIN.

Byte order The byte order in DCRAnCIN depends on the selected CRC generating function:

- 32-bit Ethernet CRC polynomial generation (DCRAnCTL.DCRAnPOL = 0)

  The byte order is least-significant-byte (LSB) first. If the CRC input bit width is 8 bits (DCRAnISZ = 10), for example, bits 7 to 0 of DCRAnCIN are the LSB.

- 16-bit CCITT CRC polynomial generation (DCRAnCTL.DCRAnPOL = 1)

  The byte order is most-significant-byte (MSB) first. If the CRC input bit width is 8 bits (DCRAnISZ = 10), for example, bits 7 to 0 of DCRAnCIN are the MSB.

Access This register can be read/written in 32-bit units.

Address <DCRAn_base>

Initial value 0000 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DCRAnCIN[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCRAnCIN[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 8-6 DCRAnCIN register contents

Bit position	Bit name	Function
31 to 0	DCRAnCIN[31:0]	Input data for CRC calculation. The valid bits are: • For 32-bit effective bit width: DCRAnCIN[31:0] • For 16-bit effective bit width: DCRAnCIN[15:0] • For 8-bit effective bit width: DCRAnCIN[7:0]

(2) DCRAnCOUT - CRC data register

This register stores the result of the CRC code generated by the 32-bit Ethernet or 16-bit CCITT polynomial.

Access This register can be read/written in 32-bit units.

Address <DCRAn_base> + 4_H

Initial value 0000 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DCRAnCOUT[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCRAnCOUT[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 8-7 DCRAnCOUT register contents

Bit position	Bit name	Function
31 to 0	DCRAnCOUT[31:0]	Result of the CRC code generation. When the 16-bit CCITT polynomial is enabled, the bits 15 to 0 show the CRC result. The bits 31 to 16 are undefined.

Caution The initial starting value must be written to this register before the first data in the data block is written to DCRAnCIN.

(3) DCRAnCTL - CRC control register

This register controls the CRC generation process.

Access This register can be read/written in 8-bit units.

Address <DCRAn_base> + 20_H

Initial value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	DCRAnISZ[1:0]		DCRAnPOL
R	R	R	R	R	R/W	R/W	R/W

Table 8-8 DCRAnCTL register contents

Bit position	Bit name	Function
2, 1	DCRAnISZ[1:0]	Specifies the CRC input bit width: 00: 32 bits (DCRAnCIN[31:0]) 01: 16 bits (DCRAnCIN[15:0]) 10: 8 bits (DCRAnCIN[7:0]) 11: Setting prohibited
0	DCRAnPOL	Specifies the CRC generating function: 0: 32-bit Ethernet CRC polynomial generation. The byte order of the DCRAnCIN register is least-significant-byte (LSB) first. If the CRC input bit width is 8 bits (DCRAnISZ = 10), for example, bits 7 to 0 of DCRAnCIN are the LSB. 1: 16-bit CCITT CRC polynomial generation. The byte order of the DCRAnCIN register is most-significant-byte (MSB) first. If the CRC input bit width is 8 bits (DCRAnISZ = 10), for example, bits 7 to 0 of DCRAnCIN are the MSB.

Note After changing the CRC generating function (DCRAnCTL.DCRAnPOL), the DCRAnCOUT register must be initialized.

Caution The CRC bit width (DCRAnCTL.DCRAnISZn) must be set according to the data block bit width. Switching the CRC bit width is not allowed during processing of a data block (a data block consists of N bytes, half words or words). The bit width can be changed after the final CRC result is read from DCRAnCOUT. At this time, specify the setting according to the operation flow shown in *Figure 8-2 “Data CRC Function A flow diagram”*.

Chapter 9 Clock Controller

This chapter describes the clock controller functions of the V850E2/Sx4-H microcontrollers.

(1) Naming rules

The clock signals and their control registers follow defined naming rules that reflect their membership to a certain power supply area and clock domain.

Index m The index m indicates the power supply area.

- m = 0: Indicates Isolated area 0 (Iso0).
- m = 1: Indicates Isolated area 1 (Iso1).
- m = A: Indicates the Always-On area (AWO).

Index n The index n (00, 01, 02, ...) indicates the clock domain.

Example The clock selector register CKSC_0n selects the clock for Isolated area 0, which is named CKSCLK_0n.
The clock signal CKSCLK_A02 is the clock supplied to clock domain 02 (n = 02) of the Always-On area (m = A). This clock is selected by way of the clock selector register CKSC_A02.

9.1 Clock Controller Overview

Overview The clock controller has the following functions:

- Four oscillators:
  - Low-speed internal oscillator with a frequency of 240 kHz (Typ.)
  - High-speed internal oscillator with a frequency of 8 MHz (Typ.)
  - Sub oscillator: 32.768 kHz
  - Main oscillator: 4 to 20 MHz
- Three PLL circuits
  - Two spread spectrum PLL circuits with adjustable frequency modulation parameters
    PLL0: Maximum 160 MHz
    PLL2: Maximum 120 MHz
  - PLL circuit with a fixed output frequency
    PLL1: Maximum 120 MHz
- Separate clock selector for each clock domain with individual standby control
- Three clock monitors (CLMA0, CLMA2, and CLMA3)
  - CLMA0: Main oscillator monitor
  - CLMA2: High-speed oscillator monitor
  - CLMA3: PLL0 monitor

Note For the specification of the clock generator frequencies, their tolerance, and other parameters, see the Data Sheet.

The following figure shows the main components of the clock controller.

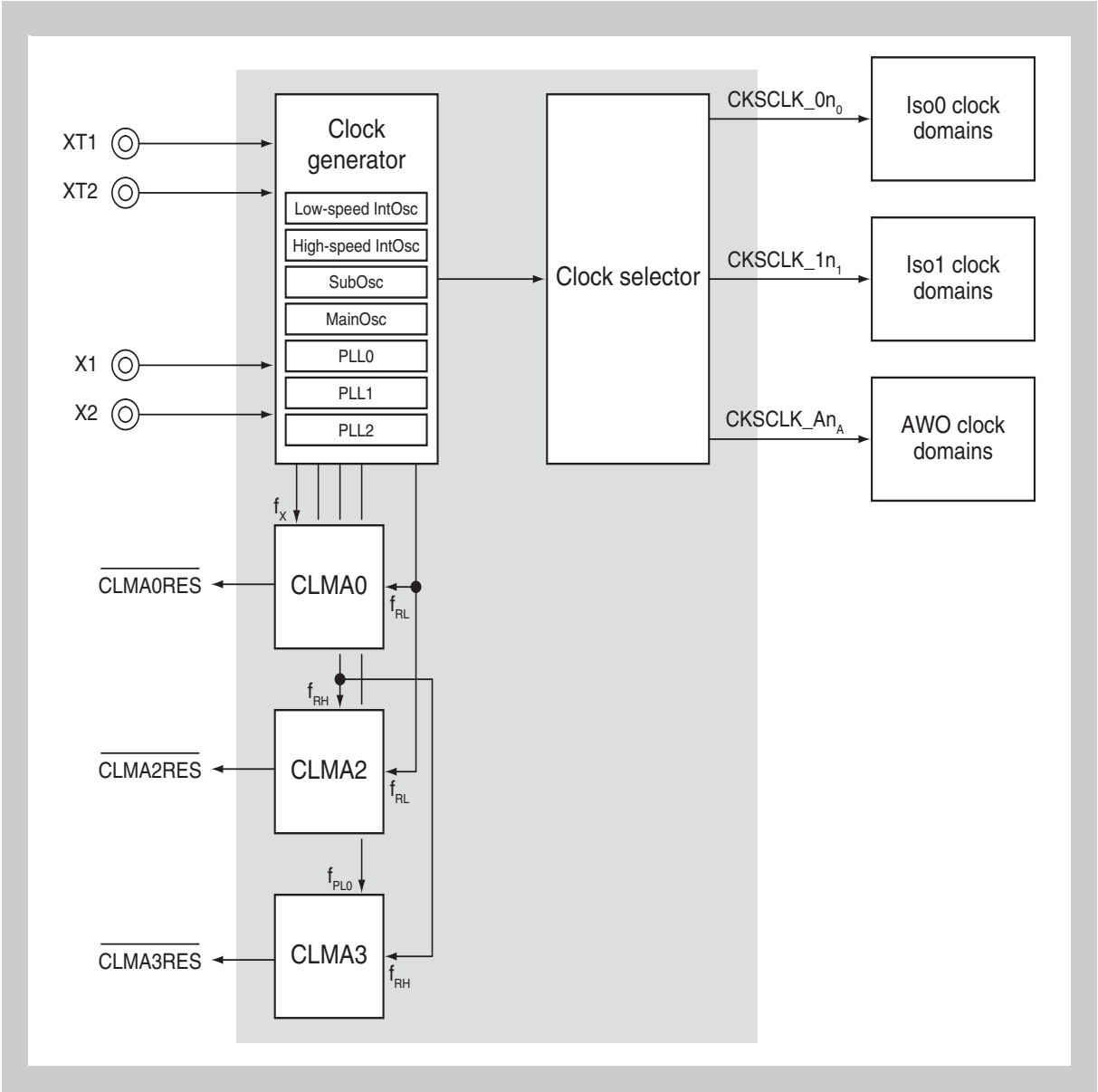


Figure 9-1 Clock controller overview

9.2 Overview of Clock Generation and Control

The clock controller generates a set of clock signals for each of the three power supply areas.

- CKSCLK_0n for clocks in Isolated area 0 (Iso0)
- CKSCLK_1n for clocks in Isolated area 1 (Iso1)
- CKSCLK_An for clocks in the Always-On area (AWO)

Each CKSCLK_mn clock signal supplies the clock for the clock domain “mn”.

All clock domains in a certain power supply area can be stopped at the same time while in the STOP standby mode by requesting that the clocks be stopped.

The following diagram outlines the basic structure of the clock controller.

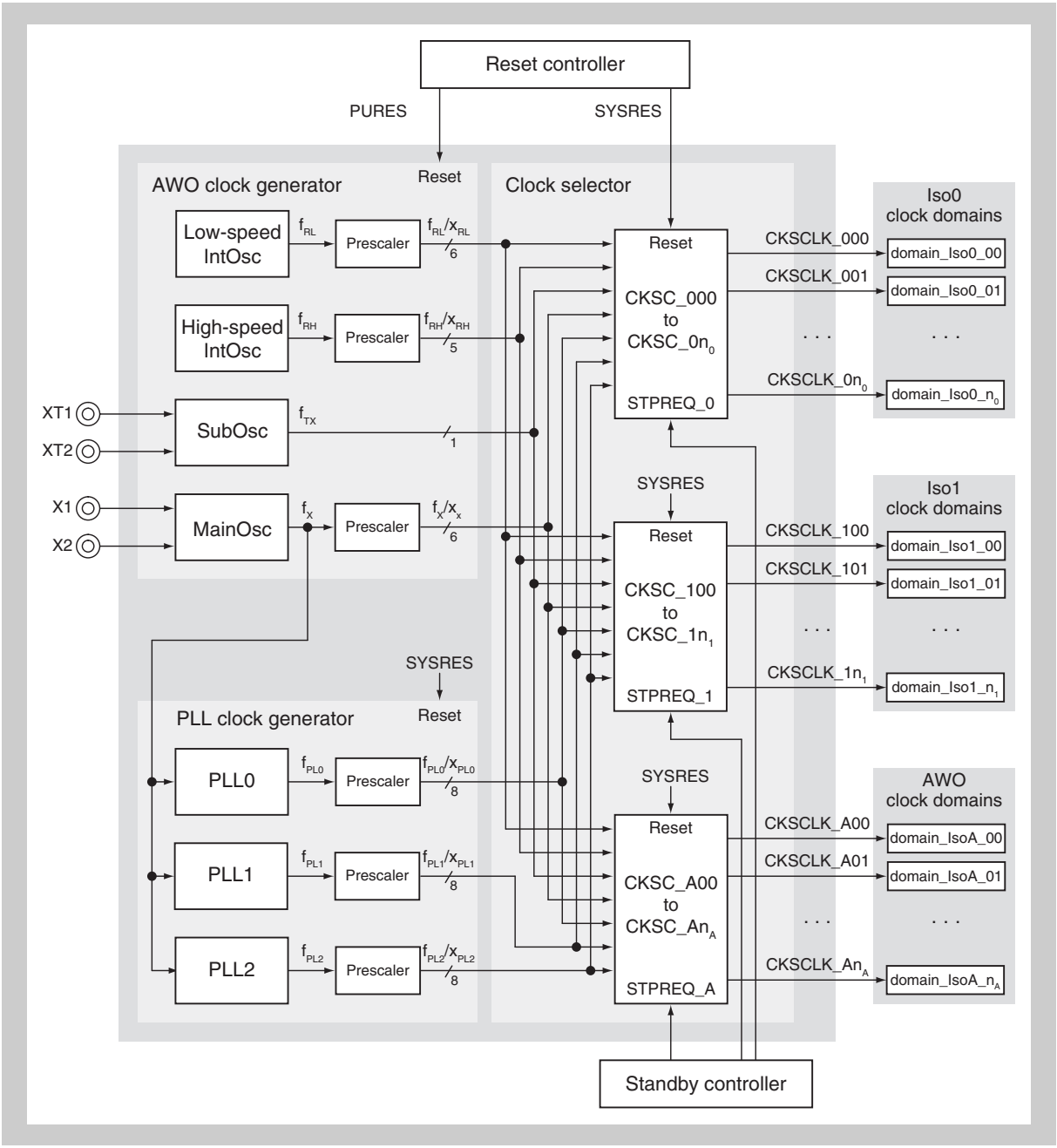


Figure 9-2 Clock controller structure

The clock controller consists of two main parts.

(1) Clock generators

Seven clock generators are provided:

- Low-speed internal oscillator (low-speed IntOsc)
  This oscillator starts operation after turning on the power and can not be stopped. Therefore, the oscillator always operates. It generates the clock f_{RL} (Typ.), which has a frequency of 240 kHz.
  This oscillator does not require any external components.
- High-speed internal oscillator (high-speed IntOsc)
  This generates the clock, which has a frequency of 8 MHz (Typ.). This oscillator does not require any external components.
- Sub oscillator (SubOsc)
  The subclock oscillator requires an external resonator (connected to XT1, XT2) to generate the subclock f_{TX}, which has a frequency of 32.768 kHz (Typ.). This clock is mainly used for applications that use a real-time clock.
- Main oscillator (MainOsc)
  The main-oscillator output clock f_X is the main system root clock and is input to the PLLs. This oscillator requires an external resonator (which is connected to X1 and X2).
- PLL0 to PLL2
  The PLL circuits generate all high-speed operation clocks (f_{PL0}, f_{PL1}, and f_{PL2}) for normal operation of the microcontroller.

<R> All the clock generators are allocated to the Always-On area (AWO).

The output clocks of the clock generators, except of the SubOsc f_{TX}, are input to prescalers, which provide different fractions of these clocks.

Note that the prescalers for all clocks have different division factors.

Clock generator reset The clock generators are initialized by the power on reset signal PURES (the power-up clear reset (for M1 products) or a reset by $\overline{\text{RESET}}$ pin input (for M2 products)). Because the clock generators are not stopped or changed by any other internal microcontroller resets, there is no need to consider the clock stabilization time after an internal reset, and the startup time can be minimized.

Fordetail about the clock generators, see *9.3 “Clock Generators” on page 431*.

Note For the specification of the clock generator frequencies, their tolerance, and other parameters, see the Data Sheet.

(2) Clock selector

The clocks generated by the clock generators are input to the clock selector CKSC_mn. A separate clock selector register (CKSC_mn) is provided for each clock domain.

The three sets of clock selector registers are dedicated to the clock domains in one of the three power supply areas:

- n_0 registers CKSC_000 to CKSC_0n_0 for clock domains in Isolated area 0
- n_1 registers CKSC_100 to CKSC_1n_1 for clock domains in Isolated area 1
- n_A registers CKSC_A00 to CKSC_An_A for clock domains in the Always-On area

By using the clock selector register CKSC_mn, one of its input clocks can be selected as the clock CKSCLK_mn.

Clock selector reset

Clock selectors are reset by using the SYSRES signal. This signal is asserted by any microcontroller POC reset or other reset. Therefore, after any reset, the initial settings are restored for supplying all clock domains.

Note that not all clocks available from the clock generators are input to each clock selector.

STOP standby mode request

The stop request for a certain power area is issued by the standby controller by asserting the corresponding stop request signal STPREQ_0 (for Iso0), STPREQ_1 (for Iso1), or STPREQ_A (for AWO) upon entering the STOP mode.

The stop request, which is supplied to all clock selectors of a power supply area, can be individually masked using the clock selector register:

- CKSC_mn.STPMK_mn bit = 0:
  STPREQ_m is not masked, so CLSCLK_mn is stopped if STPREQ_m is asserted.
- CKSC_mn.STPMK_mn bit = 1:
  STPREQ_m is masked, so CLSCLK_mn remains in operation even if STPREQ_m is asserted.

See *9.3 “Clock Generators” on page 431* for a detailed description of the clock generators.

9.3 Clock Generators

9.3.1 Main oscillator (MainOsc) clock generator

The main oscillator generates the clock f_X, which is supplied to the clock domain clock selector CKSC_m. f_X is also used as the PLL input clock PLLCLKIN.

The diagram below shows the basic structure and signals of the MainOsc clock generator.

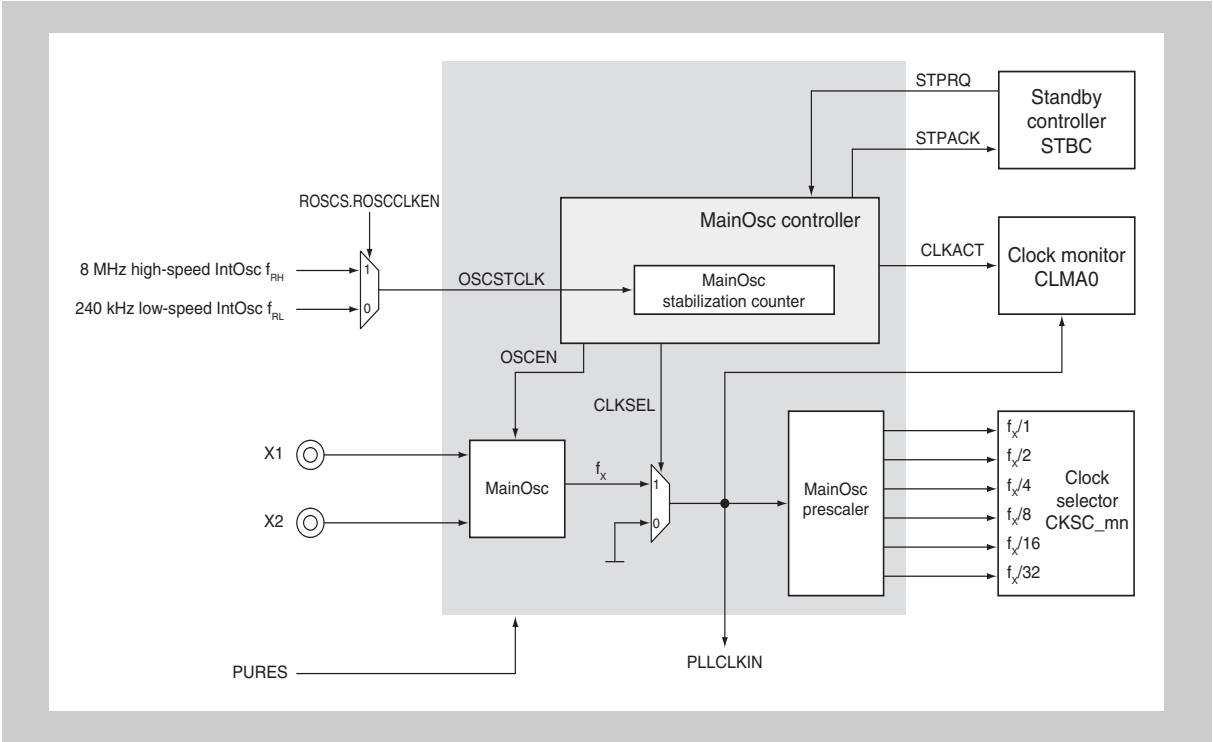


Figure 9-3 Main oscillator clock generator

MainOsc enable

After a reset ends, the MainOsc is disabled and must therefore be enabled by setting the MainOsc enable trigger bit MOSCE.ENTRG to 1.
In addition, when waking up from the standby mode, the MainOsc can be started by using the standby cancellation source.
The MainOsc enabled status is indicated by the bit MOSCS.MOSCCLKEN being 1.

MainOsc stabilization

The MainOsc stabilization counter is used to decrement the oscillator stabilization time.

While the MainOsc is not stable, the CLKSEL signal disables the f_X output to the MainOsc prescaler.
If the MainOsc stabilization counter has reached the value defined by the MOSCST.MOST[3:0] bits, f_{TX} is judged as stable, the CLKSEL signal value is changed, and f_X is input to the prescaler. This makes it possible to select the prescaler output $f_X/1$, $f_X/2$, $f_X/4$, $f_X/8$, $f_X/16$, or $f_X/64$ by using the clock selector CKSC_mn.
The f_X clock stable status is indicated by the bit MOSCS.MOSCCLKSTAB being 1.

The stabilization counter clock OSCSTCLK is selected from two sources:

- OSCSTCLK = the 8 MHz clock f_{RH} if the high-speed internal oscillator is operating (the ROSCS.ROSCCLKEN bit = 1)
- OSCSTCLK = the 240 kHz low-speed internal oscillator clock f_{RL} if the high-speed internal oscillator is disabled (ROSCS.ROCCCLKEN = 0)

The stabilization counter clock source is selected automatically according to the high-speed internal oscillator operation status.

The MOSCST.MOST[3:0] bits determine the main oscillator stabilization time as a number of OSCSTCLK periods. The setting range is 2^2 to 2^{17}.

MainOsc amplification gain

The MainOsc input frequency from the external resonator can be selected by using the MOSCC.AMPSEL[1:0] bits in the range from 4 to 20 MHz.
The MOSCC.AMPSEL[1:0] bits adjust the oscillator's amplifier gain to different input frequency ranges to optimize the stabilization time.
Depending on the external resonator's operation and the external circuitry, the amplification gain can be forced to its maximum, thus disregarding the MOSCC.AMPSEL[1:0] bits, by setting the MOSCC.SHTSTBY bit to 1. This might result in a shorter oscillator stabilization time.

STOP standby mode request

By using the STPRQ signal, the standby controller indicates the STOP standby status. The stop request mask bit MOSCE.MOSCSTPMK controls whether the MainOsc is stopped while in the STOP standby mode or continues operation.
If the MainOsc is stopped while in the STOP standby mode (MOSCE.MOSCSTPMK bit = 0), it is automatically re-started upon waking up from the mode.
A MainOsc standby stop is acknowledged to the standby controller by way of the signal STPACK, which is also applied to the MOSCS.MOSCSTPACK bit.

Clock monitor control

The MainOsc activity signal CLKACT is output to the clock monitor CLMA0 to control its operation. If the MainOsc is inactive, the monitoring of its output clock f_X by CLMA0 is also deactivated.

The table below summarizes the various conditions for the clock monitor control.

Table 9-1 Clock monitor 0 status control

MOSCS. MOSCCLKEN	MOSCE. MOSCSTPMK	STPRQ	CLKACT	CLMA0
0	X	X	0	Stopped
1	0	0	1	Active
		1	0	Stopped
	1	X	1	Active

MainOsc operation/ stop trigger

The MainOsc can be operated or stopped using the triggers below.

- The MainOsc is started by setting the operation enable trigger bit MOSCE.MOSCENTRG to 1.
  The operation enable trigger is only valid when the MainOsc is inactive (when the MOSCS.MOSCCLKACT bit = 0).
- The MainOsc is stopped by setting the operation stop trigger bit MOSCE.MOSCDISTRG to 1.
  The operation stop trigger is only valid when the MainOsc is active (when the MOSCS.MOSCCLKACT bit = 1).

9.3.2 Sub oscillator (SubOsc) clock generator

The sub oscillator generates the sub clock f_{TX}, which is supplied to the clock domain clock selector CKSC_mn. f_{TX} normally has a frequency of 32.768 kHz (Typ.) and is mainly used as for the real-time clock.

The diagram below shows the basic structure and signals of the SubOsc clock generator.

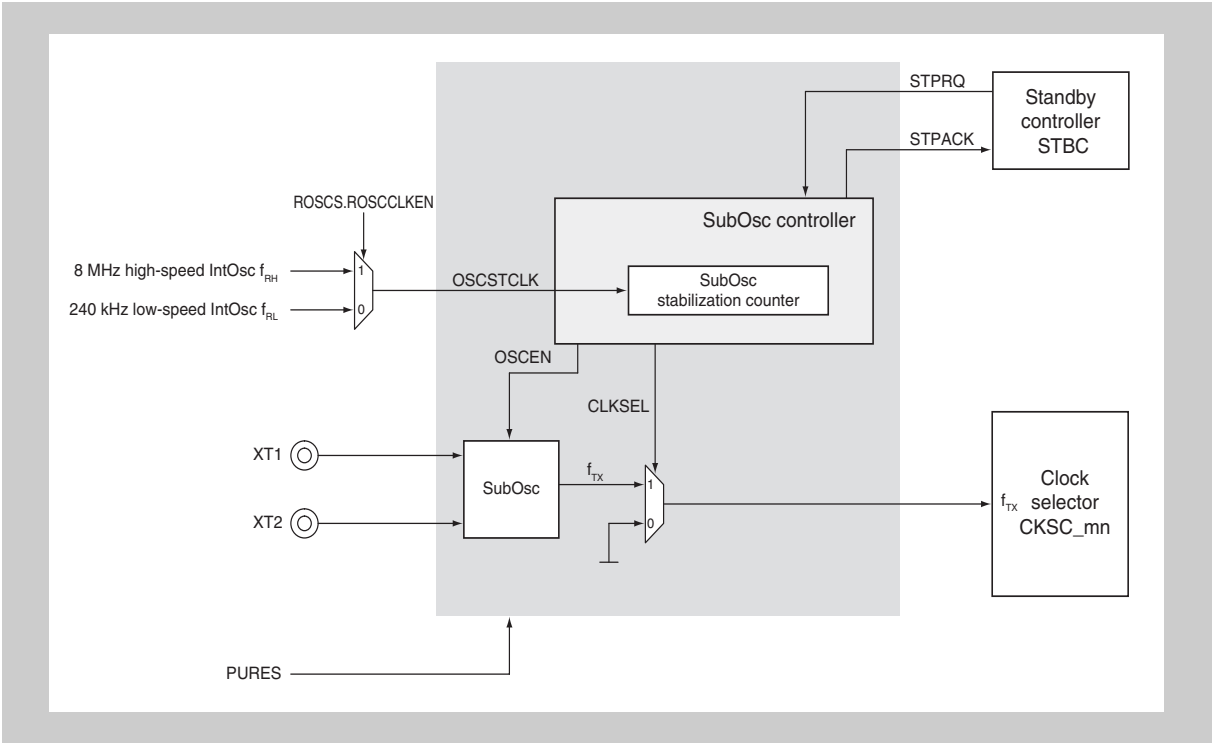


Figure 9-4 Sub oscillator clock generator

SubOsc enable

After a reset ends, the SubOsc is disabled and must therefore be enabled by setting the SubOsc enable trigger bit SOSCE.SOSCENTRG to 1.
The SubOsc enabled status is indicated by the bit SOSCS.SOSCCLKEN being 1.

SubOsc stabilization

The SubOsc stabilization counter is used to decrement the stabilization time.

While the SubOsc is not stable, the CLKSEL signal disables the f_{TX} output to the clock selector CKSC_mn.
If the SubOsc stabilization counter has reached the value defined by the SOSCST.SOST[2:0] bits, f_{TX} is judged as stable, the CLKSEL signal value is changed, and f_{TX} is input to the clock selector CKSC_mn. The f_{TX} clock stable status is indicated by the bit SOSCS.SOSCCLKSTAB being 1.

The stabilization counter clock OSCSTCLK is selected from the following two sources:

- OSCSTCLK = the 8 MHz clock f_{RH} if the high-speed internal oscillator is operating (the ROSCS.ROSCCLKEN bit = 1)
- OSCSTCLK = the 240 kHz low-speed internal oscillator clock f_{RL} if the high-speed internal oscillator is disabled (ROSCS.ROCCCLKEN = 0)

The stabilization counter clock source is selected automatically according to the high-speed internal oscillator operation status.

The SOSCST.SOST[2:0] bits determine the sub oscillator stabilization time as a number of OSCSTCLK periods. The setting range is 2^{19} to 2^{26}.

SubOsc input frequencies

The SubOsc input frequency is typically 32.768 kHz.

STOP standby mode request

By using the STPRQ signal, the standby controller indicates the STOP standby status. The stop request mask bit SOSCE.SOSCSTPMK controls whether the SubOsc is stopped while in the STOP standby mode or continues operation.
If the SubOsc is stopped while in the STOP standby mode (SOSCE.SOSCSTPMK bit = 0), it is not automatically restarted even after waking up from the mode. To restart the SubOsc, use software.
A SubOsc standby stop is acknowledged to the standby controller by way of the signal STPACK, which is also applied to the SOSCS.SOSCSTPACK bit.

SubOsc operation/ stop trigger

The SubOsc can be operated or stopped using the triggers below.

- The SubOsc is started by setting the operation enable trigger bit SOSCE.SOSCENTRG to 1.
  The operation enable trigger is only valid when the SubOsc is inactive (when the SOSCS.SOSCCLKACT bit = 0).
- The SubOsc is stopped by setting the operation stop trigger bit SOSCE.SOSCDISTRG to 1.
  The operation stop trigger is only valid when the SubOsc is active (when the SOSCS.SOSCCLKACT bit = 1).

<R>

9.3.3 High-speed internal oscillator (high-speed IntOsc) clock generator

The high-speed internal oscillator generates the clock f_{RH}, which is supplied to the clock domain clock selector CKSC_mn. f_{RH} has a nominal frequency of 8 MHz (Typ.).

The diagram below shows the basic structure and signals of the high-speed IntOsc clock generator.

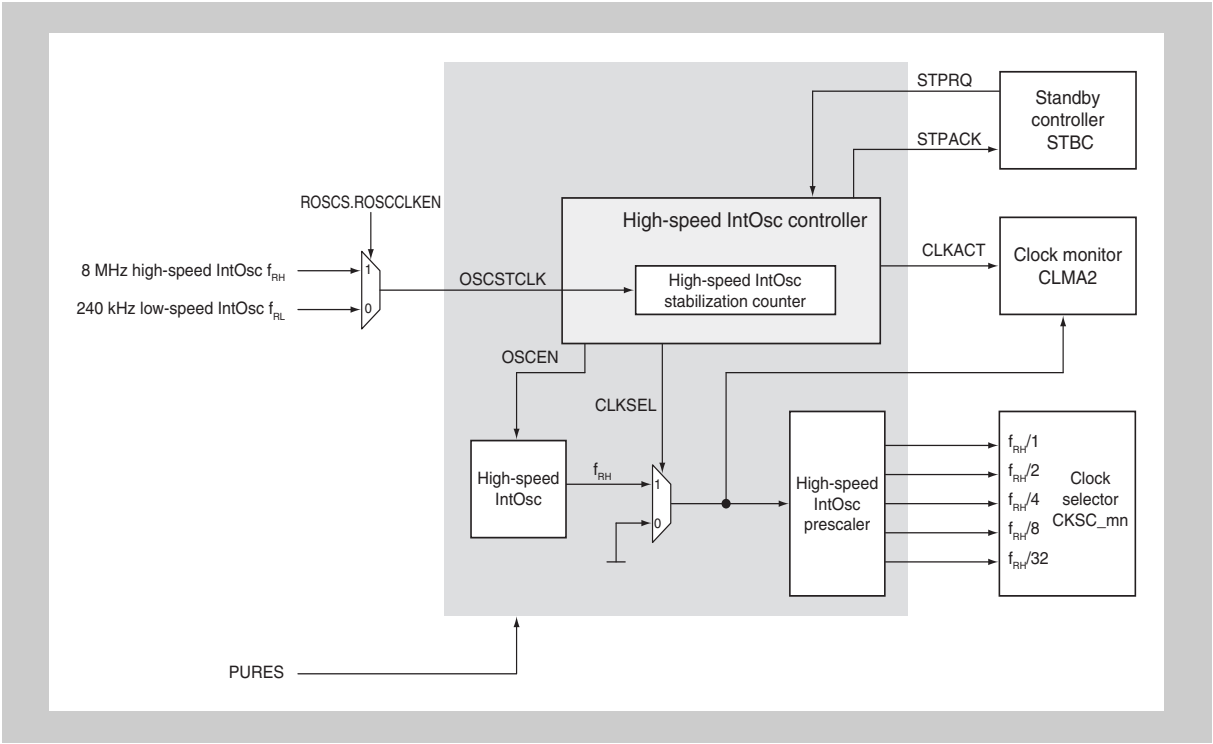


Figure 9-5 High-speed internal oscillator clock generator

After a reset ends, the high-speed IntOsc starts operation.

STOP standby mode request

By using the STPRQ signal, the standby controller indicates the STOP standby status. The stop request mask bit ROSCE.ROSCSTPMK controls whether the high-speed IntOsc is stopped during STOP standby or continues operation.
If the high-speed IntOsc is stopped during STOP standby (ROSCE.ROSCSTPMK = 0), it is automatically re-started upon wake-up from the STOP standby mode.
A high-speed IntOsc standby stop is acknowledged to the standby controller by way of the signal STPACK, which is also applied to the ROSCS.ROSCSTPACK bit.

Clock monitor control

The high-speed IntOsc activity signal CLKACT is output to the clock monitor CLMA2 to control its operation. If the high-speed IntOsc is inactive, the monitoring of its output clock f_{RH} by CLMA2 is also deactivated.

The table below summarizes the various conditions for the clock monitor control.

Table 9-2 Clock monitor 2 status control

ROSCS. ROSCCLKEN	ROSCE. ROSCSTPMK	STPRQ	CLKACT	CLMA2
0	X	X	0	Stopped
1	0	0	1	Active
		1	0	Stopped
	1	X	1	Active

The high-speed IntOsc clock f_{RH} is used as the sampling clock for the clock monitor CLMA3.

9.3.4 Low-speed internal oscillator (low-speed IntOsc) clock generator

The low-speed internal oscillator generates the clock f_{RL}, which is supplied to the clock domain clock selector CKSC_mn. f_{RL} has a frequency of 240 kHz (Typ.).

The diagram below shows the basic structure and signals of the low-speed IntOsc clock generator.

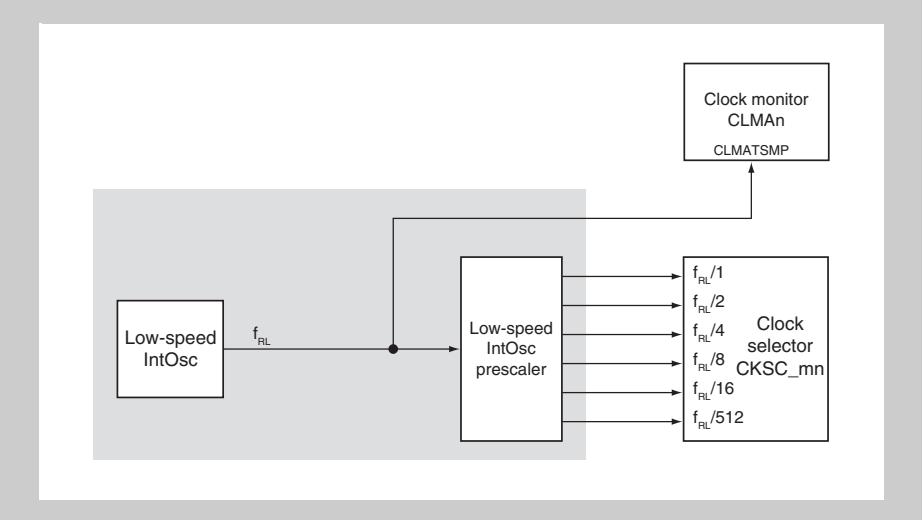


Figure 9-6 Low-speed internal oscillator clock generator

After a reset ends, the low-speed IntOsc starts operation. It cannot be stopped.

The low-speed IntOsc clock f_{RL} is used as the sampling clock for the clock monitors CLMA0 and CLMA2.

9.3.5 Phase-locked loop (PLL) clock generators

The main oscillator clock f_X is input to the phase-locked loop clock generator PLLk. The PLLk output clock f_{PLk} is a multiple of f_X and serves as the main operation clock for the microcontroller.

The diagram below shows the basic structure and signals of the PLLk clock generator.

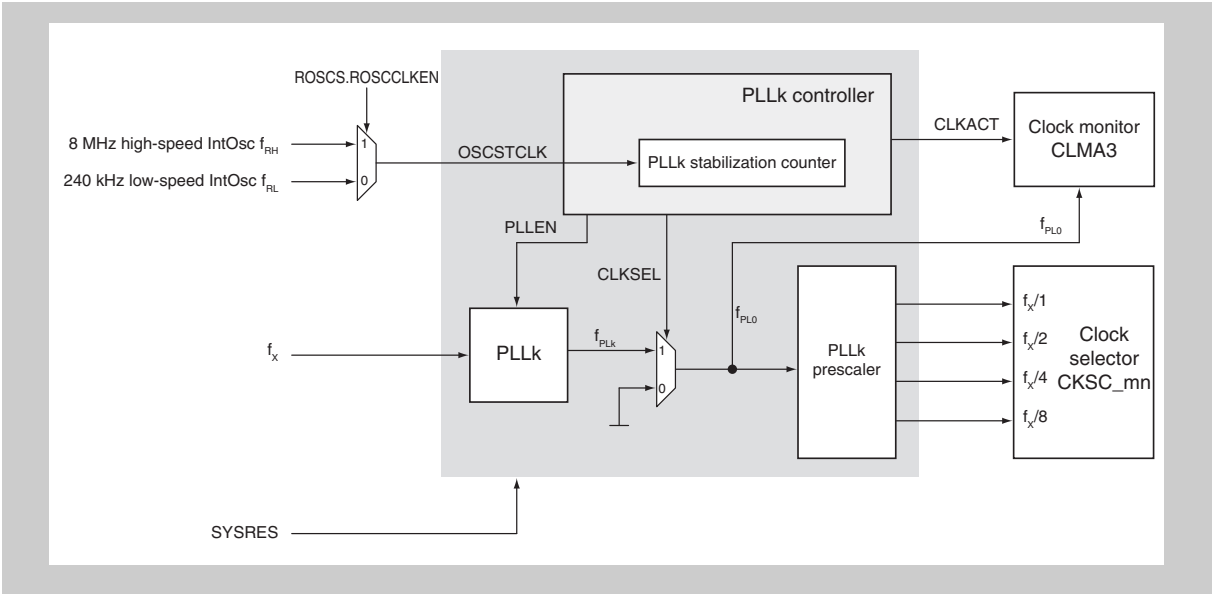


Figure 9-7 PLLk clock generator

PLLk enable

After a reset ends, PLLk is disabled and must therefore be enabled by setting the PLLk enable trigger bit PLLEk.PLLkENTRG to 1. The PLLk enabled status is indicated by the bit PLLSk.PLLkCLKENk being 1.

PLLk stabilization

The PLLk stabilization counter is used to decrement the stabilization time.

While the PLLk is not stable, the CLKSEL signal disables the f_{PLk} output to the PLLk prescaler.
If the PLLk stabilization counter has reached the value defined by the PLLSTk.PLLSTk[2:0] bits, f_{PLk} is judged as stable, the CLKSEL signal value is changed, and f_X is input to the prescaler. This makes it possible to select the prescaler output $f_{PLk}/1$, $f_{PLk}/2$, $f_{PLk}/4$, or $f_{PLk}/8$ by using the clock selector CKSC_mn.
The f_{PLk} clock stable status is indicated by the bit PLLSk.PLLkCLKSTAB being 1.

The stabilization counter clock OSCSTCLK is selected from the following two sources:

- OSCSTCLK = the 8 MHz clock f_{RH} if the high-speed internal oscillator is operating (the ROSCS.ROSCCLKEN bit = 1)
- OSCSTCLK = the 240 kHz low-speed internal oscillator clock f_{RL} if the high-speed internal oscillator is disabled (ROSCS.ROCCCLKEN = 0)

The stabilization counter clock source is selected automatically according to the high-speed internal oscillator operation status.

The PLLSTk.PLLSTk[2:0] bits determine the PLLk stabilization time as a number of OSCSTCLK periods. The setting range is 2^7 to 2^{14}.

STOP standby mode request

By using the STPRQ signal, the standby controller indicates the STOP standby status. The stop request mask bit PLLEk.PLLkSTPMK controls whether the PLLk is stopped during STOP standby or continues operation.
If PLLk is stopped while in the STOP standby mode (PLLEk.PLLkSTPMK bit = 0), it is automatically restarted upon waking up from the mode.
A PLLk stop is acknowledged to the standby controller by way of the signal STPACK, which is also applied to the PLLSk.PLLkSTPACK bit.

Clock monitor control

The PLL0 activity signal CLKACT is output to the clock monitor CLMA3 to control its operation. If PLL0 is inactive, the monitoring of its output clock f_{PL0} by CLMA3 is also deactivated.

The table below summarizes the various conditions for the clock monitor control.

Table 9-3 Clock monitor 3 status control

PLLSk. PLLkCLKEN	PLLEk. PLLkSTPMK	STPRQ	CLKACT	CLMA3
0	X	X	0	Stopped
1	0	0	1	Active
		1	0	Stopped
	1	X	1	Active

PLLk operation/ stop trigger

PLLk can be operated or stopped using the triggers below.

- PLLk is started by setting the operation enable trigger bit PLLEk.PLLkENTRG to 1.
  The operation enable trigger is only valid when PLLk is inactive (when the PLLSk.PLLkCLKACT bit = 0).
- PLLk is stopped by setting the operation stop trigger bit PLLSk.PLLkCLKACT to 1.
  The operation stop trigger is only valid when PLLk is active (when the PLLSk.PLLkCLKACT bit = 1).

(1) PLL parameters

The PLL is configured by a set of parameters, derived from the control register PLLCk.

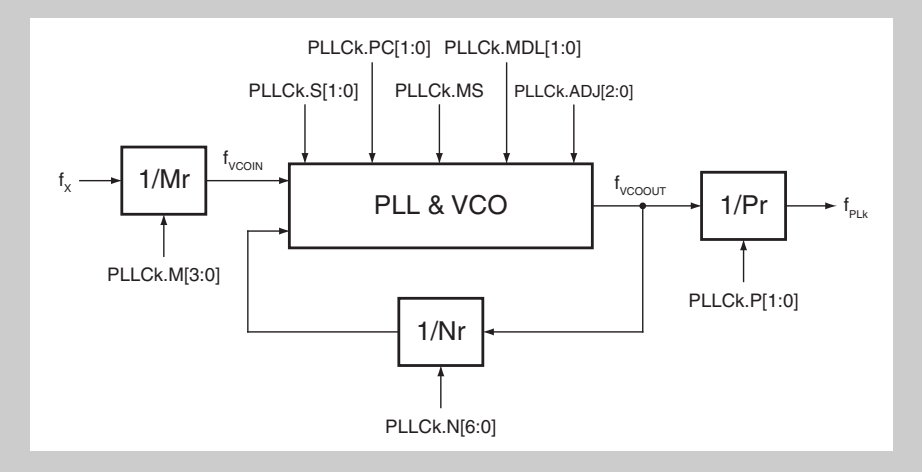


Figure 9-8 PLL circuit

PLL mode

The PLL mode is selected by from the following two modes:

- PLLCk.MS bit = 0:
  SSCG (Spread Spectrum Clock Generator) mode with a modulated output frequency
- PLLCk.MS bit = 1: PLL mode with a fixed output frequency

f_{PLk}

The frequency of f_{PLk} is calculated as follows:

$$f_{PLk} = f_X \times (Nr/Mr) \times 1/Pr$$

The values Nr, Mr, and Pr are derived from PLLCk register bits:

- Nr = PLLCk.N[6:0] + 1
- The Mr value is calculated based on which of the following two modes is specified:
  - SSCG mode (PLLCk.MS bit = 0): Mr = PLLCk.M[3:0] + 1
  - PLL mode (PLLCk.MS bit = 1): Mr = 1 (PLLCk.M[3:0] bit = 0H)
- Pr is determined according to the PLLCk.P[1:0] bits as follows:

PLLCk.P[1:0]	Pr
00_B	0.5
01_B	1
10_B	2
11_B	4

Frequency modulation

If frequency dithering is enabled (the PLLCk.MS bit = 1), additional parameters must be set in PLLCk:

- The PLLCk.PC[1:0] bits determine the dithering mode
  - PC[1:0] = $0X_B$: Fixed output frequency
  - PC[1:0] = 10_B: Down spread mode (maximum output frequency)
  - PC[1:0] = 11_B: Center spread mode (center output frequency)

- PLLCk.MDL[1:0] determines the frequency control period (the time the output frequency changes from its maximum to its minimum and vice versa).
- The PLLCk.ADJ[2:0] bits determine the frequency modulation range (the maximum and minimum frequency).

For details about the allowed settings of the PLL parameters, see the description of the PLLCk register.

PLLk input frequencies

The VCO input frequency f_{VCOIN} must lie in the range from 1 to 2 MHz (only in the SSCG mode). Therefore, the division factor Mr must be set correctly, as follows:

$f_{VCOIN} = f_X/Mr$ = 1 to 2 MHz.

The PLLCk.S[1:0] bit must be set in accordance with the frequency f_{VCOIN}.

9.3.6 Writing to protected registers

Write protected registers are protected from inadvertent write access due to erroneous program execution or other causes.

Writing to a write protected register is only possible through the following special instruction sequence:

1. Write the fixed value $A5_H$ to the protection command register PROTCMDm.
2. Write the desired value to the protected register.
3. Write the bit-wise inversion of the desired value to the protected register.
4. Write the desired value to the protected register.
5. Verify that the desired value was successfully written to the protected register by verifying that the PROTSm.PROTERR bit = 0. If the confirmation result is 1, it is necessary to restart from step 1.

If there is write access to the clock controller registers during step 1 to step 4 of the above sequence, the write to the protected register fails (indicated by PROTSm.PROTERR = 1). In this case, it is necessary to restart the instruction sequence from step 1.

Within the special instruction sequence, it is allowed to access other registers, except the clock controller registers, without disrupting the protection sequence.

If the protection sequence is interrupted, the protection mechanism operates as follows:

- Interrupts during the protection sequence
  If an interrupt is acknowledged during the above protection sequence and the interrupt service routine is not accessing any clock controller registers, the protection sequence is not disrupted. After processing returns from the interrupt service routine, the protected register is written to normally.
- Emulator break during the protection sequence
  If an emulation break occurs during the above protection sequence, such as due to a breakpoint hit, the register protection is suspended until normal operation is resumed after the break.
  This means that, even if a clock controller register is accessed during the break, the protection sequence is not disrupted.
  Access to the clock controller registers does not set the PROTSm.PROTERR bit.

9.4 Clock Selection

This section describes all clock selection options for all clock domains in the three power areas of the V850E2/Sx4-H products.

The names of clock selector control and status registers use two indices to identify the power supply area and clock domain:

m = 0: Isolated area 0

If CKSC_mn controls the clock of any clock domain within Isolated area 1:

- m = 0
- n = 00, 05 to 07, 11, 12, 16

m = 1: Isolated area 1

If CKSC_mn controls the clock of any clock domain within Isolated area 1:

- m = 1:
- n = 01 to 03, 05 to 17, 22 to 24, 28, 29

m = A: Always-On area

If CKSC_mn controls the clock of any clock domain in the Always-On area:

- m = A:
- n = 02, 03, 05, 07, 09

For each clock selector register, a separate table is provided that provides information about:

- Power supply areas and clock domains
- Clock selector register name, address, and initial value
- The domain clock name
- The clock selection options, their ID, and clock limitations (if applicable)

9.4.1 Clock domains of the Always-On area

(1) Clock domain AWO_2

Clock selector control register: CKSC_A02		Power supply area: Always-On area		
Address	$FF42\ 2020_H$	Clock domain: AWO_2		
Initialvalue	$0000\ 000E_H$			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0001_H	Low-speed IntOsc (240 kHz) / 1	–	CKSCLK_A02	RTCA0: PCLK WDTA0: PCLK CLMA0: PCLK CLMA1: PCLK
0007_H	High-speed IntOsc (8 MHz) / 1			
0008_H	High-speed IntOsc (8 MHz) / 2			
0009_H	High-speed IntOsc (8 MHz) / 4			
$000A_H$	High-speed IntOsc (8 MHz) / 8			
$001C_H$	PLL1 / 1	≤ 80 MHz		
$001D_H$	PLL1 / 2	≤ 60 MHz		
$001F_H$	PLL1 / 4	–		
0022_H	PLL1 / 8			
0000_H	No clock selected			
Other than the above	Setting prohibited			

<R> **Caution** The clock source selected by the above CKSC_mn register is used by the entire system. Therefore, be sure not to stop the clock source by using software.

(2) Clock domain AWO_3

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_A03		**Power supply area: Always-On area**		
Address	**FF42 2030$_H$**	**Clock domain: AWO_3**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0001$_H$	Low-speed IntOsc (240 kHz) / 1	–	CKSCLK_A03	TAUJ0: PCLK
0007$_H$	High-speed IntOsc (8 MHz) / 1			
000C$_H$	MainOsc / 1			
0012$_H$	SubOsc [32 kHz]			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(3) Clock domain AWO_5

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_A05		**Power supply area: Always-On area**		
Address	**FF42 2050$_H$**	**Clock domain: AWO_5**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_A05	Backup RAM: PCLK
001C$_H$	PLL1 / 1	≤ 40 MHz		
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4	≤ 30 MHz		
0022$_H$	PLL1 / 8	–		
Other than the above	Setting prohibited			

(4) Clock domain AWO_7

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_A07		**Power supply area: Always-On area**		
Address	**FF42 2070$_H$**	**Clock domain: AWO_7**		
Initial value	**0000 0006$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0001$_H$	Low-speed IntOsc (240 kHz) / 1	–	CKSCLK_A07	WDTA0: WDTACKI
0003$_H$	Low-speed IntOsc (240 kHz) / 4			
0005$_H$	Low-speed IntOsc (240 kHz) / 512			
Other than the above	Setting prohibited			

(5) Clock domain AWO_9

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_A09		**Power supply area: Always-On area**		
Address	**FF42 2090$_H$**	**Clock domain: AWO_9**		
Initial value	**0000 0000$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0009$_H$	High-speed IntOsc (8 MHz) / 4	–	CKSCLK_A09	RTCA0: RTCATCKI
000A$_H$	High-speed IntOsc (8 MHz) / 8			
000C$_H$	MainOsc / 1			
000D$_H$	MainOsc / 2			
000E$_H$	MainOsc / 4			
000F$_H$	MainOsc / 8			
0012$_H$	SubOsc (32 kHz)			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

9.4.2 Clock domains of Isolated area 0

(1) Clock domain ISO0_0

Clock selector control register: CKSC_000		Power supply area: Isolated area 0		
Address	$FF42\ 6000_H$	Clock domain: ISO0_0		
Initial value	$0000\ 0074_H$			
Clock source ID	Clock source	Clock limitation	Domain clock	Module: Clock
0008_H	High-speed IntOsc (8 MHz) / 2	–	CKSCLK_000	CPU, CPU subsystem
0009_H	High-speed IntOsc (8 MHz) / 4			
$000A_H$	High-speed IntOsc (8 MHz) / 8			
$000B_H$	High-speed IntOsc (8 MHz) / 32			
$000C_H$	MainOsc / 1			
0014_H	PLL0 / 1	≤ 160 MHz		
0015_H	PLL0 / 2	–		
0017_H	PLL0 / 4			
$001A_H$	PLL0 / 8			
$003A_H$	High-speed IntOsc (8 MHz) (Low-speed IntOsc (240 kHz))[a]			
Other than the above	Setting prohibited			

a) If the high-speed IntOsc is disabled, the low-speed IntOsc is automatically selected.

<R> **Caution** The clock source selected by the above CKSC_mn register is used by the entire system. Therefore, be sure not to stop the clock source by using software.

(2) Clock domain ISO0_5

Clock selector control register: CKSC_005		Power supply area: Isolated area 0		
Address	FF42 6050$_H$	Clock domain: ISO0_5		
Initial value	0000 00E$_H$			
Clock source ID	Clock source	Clock limitation	Domain clock	Module: Clock
0001$_H$	Low-speed IntOsc (240 kHz) / 1	–	CKSCLK_005	WDTA1: PCLK DNFAn: PCLK
0007$_H$	High-speed IntOsc (8 MHz) / 1			
0008$_H$	High-speed IntOsc (8 MHz) / 2			
0009$_H$	High-speed IntOsc (8 MHz) / 4			
000A$_H$	High-speed IntOsc (8 MHz) / 8			
0014$_H$	PLL0 / 1	≤ 80 MHz		
0015$_H$	PLL0 / 2			
0017$_H$	PLL0 / 4	–		
001A$_H$	PLL0 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

<R> **Caution** The clock source selected by the above CKSC_mn register is used by the entire system. Therefore, be sure not to stop the clock source by using software.

(3) Clock domain ISO0_6

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_006		**Power supply area: Isolated area 0**		
Address	**FF42 6060$_H$**	**Clock domain: ISO0_6**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_006	TAUA0: PCLK ENCA0: PCLK ENCA1: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(4) Clock domain ISO0_7

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_007		**Power supply area: Isolated area 0**		
Address	**FF42 6070$_H$**	**Clock domain: ISO0_7**		
Initial value	**0000 0006$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0001$_H$	Low-speed IntOsc (240 kHz) / 1	–	CKSCLK_007	WDTA1: WDTATCKI
0003$_H$	Low-speed IntOsc (240 kHz) / 4			
Other than the above	Setting prohibited			

(5) Clock domain ISO0_11

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_011		**Power supply area: Isolated area 0**		
Address	**FF42 60B0$_H$**	**Clock domain: ISO0_11**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_011	URTE10: PCLK LMA10: PCLK CNTA2: PCLK CSIG4: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(6) Clock domain ISO0_12

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_012		**Power supply area: Isolated area 0**		
Address	**FF42 60C0$_H$**	**Clock domain: ISO0_12**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_012	ADCA0: PCLK
000C$_H$	MainOsc			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(7) Clock domain ISO0_16

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_016		**Power supply area: Isolated area 0**		
Initial address value	**FF42 6100$_H$ 0000 000E$_H$**	**Clock domain: ISO0_16**		
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_016	DNFAn: DNFATCKI
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
Other than the above	Setting prohibited			

9.4.3 Clock domains of Isolated area 1

(1) Clock domain ISO1_1

Clock selector control register: CKSC_101		Power supply area: Isolated area 1		
Address	$FF42\ A010_H$	Clock domain: ISO1_1		
Initial value	$0000\ 000E_H$			
Clock source ID	Clock source	Clock limitation	Domain clock	Module: Clock
0001_H	Low-speed IntOsc (240 kHz) / 1	–	CKSCLK_101	DNFAn: PCLK DCRA: PCLK
0007_H	High-speed IntOsc (8 MHz) / 1			
0008_H	High-speed IntOsc (8 MHz) / 2			
0009_H	High-speed IntOsc (8 MHz) / 4			
$000A_H$	High-speed IntOsc (8 MHz) / 8			
0014_H	PLL0 / 1	≤ 80 MHz		
0015_H	PLL0 / 2	≤ 60 MHz		
0017_H	PLL0 / 4	–		
$001A_H$	PLL0 / 8			
0000_H	No clock selected			
Other than the above	Setting prohibited			

<R> **Caution** The clock source selected by the above CKSC_mn register is used by the entire system. Therefore, be sure not to stop the clock source by using software.

(2) Clock domain ISO1_2

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_102		**Power supply area: Isolated area 1**		
Address	**FF42 A020$_H$**	**Clock domain: ISO1_2**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_102	IISA0: PCLK IISAn and PCMn clock divider (n = 0 to 5): PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(3) Clock domain ISO1_3

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_103		**Power supply area: Isolated area 1**		
Address	**FF42 A030$_H$**	**Clock domain: ISO1_3**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_103	IISA1: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(4) Clock domain ISO1_5

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_105		**Power supply area: Isolated area 1**		
Address	**FF42 A050$_H$**	**Clock domain: ISO1_5**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_105	IISA2: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(5) Clock domain ISO1_6

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_106		**Power supply area: Isolated area 1**		
Address	**FF42 A060$_H$**	**Clock domain: ISO1_6**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_106	IEBB0: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(6) Clock domain ISO1_7

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_107		**Power supply area: Isolated area 1**		
Address	**FF42 A070$_H$**	**Clock domain: ISO1_7**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_107	IICB2: PCLK IICB3: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(7) Clock domain ISO1_8

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_108		**Power supply area: Isolated area 1**		
Address	**FF42 A080$_H$**	**Clock domain: ISO1_8**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_108	CSIG0: PCLK IICB0: PCLK IICB1: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(8) Clock domain ISO1_9

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_109		**Power supply area: Isolated area 1**		
Address	**FF42 A090$_H$**	**Clock domain: ISO1_9**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_109	IISA3: PCLK CSIH0: PCLK CSIH1: PCLK CSIH2: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(9) Clock domain ISO1_10

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_110		**Power supply area: Isolated area 1**		
Address	**FF42 A0A0$_H$**	**Clock domain: ISO1_10**		
Initial value	**0000 0000$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_110	IISABRG00: IISABRG00SCK
0014$_H$	PLL0 / 1	≤ 160 MHz		
0015$_H$	PLL0 / 2	–		
0017$_H$	PLL0 / 4			
001A$_H$	PLL0 / 8			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
003D$_H$	External IISA clock IISAACK			
003E$_H$	External MLB clock MLBA0CLK			
0000H	No clock selected			
Other than the above	Setting prohibited			

(10) Clock domain ISO1_11

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_111		**Power supply area: Isolated area 1**		
Address	**FF42 A0B0$_H$**	**Clock domain: ISO1_11**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_111	TAUB2: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(11) Clock domain ISO1_12

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_112		**Power supply area: Isolated area 1**		
Address	**FF42 A0C0$_H$**	**Clock domain: ISO1_12**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_112	URTE0: PCLK URTE1: PCLK LMA0: PCLK LMA1: PCLK CNTA0: PCLK OSTM0: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(12) Clock domain ISO1_13

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_113		**Power supply area: Isolated area 1**		
Address	**FF42 A0D0$_H$**	**Clock domain: ISO1_13**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_113	FCN0: PCLK FCN1: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(13) Clock domain ISO1_14

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_114		**Power supply area: Isolated area 1**		
Address	**FF42 A0E0$_H$**	**Clock domain: ISO1_14**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_114	URTE2: PCLK URTE3: PCLK CNTA1: PCLK LMA2: PCLK LMA3: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(14) Clock domain ISO1_15

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_115		**Power supply area: Isolated area 1**		
Address	**FF42 A0F0$_H$**	**Clock domain: ISO1_15**		
Initial value	**0000 0000$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_115	IISABRG10: IISABRG10SCK
0014$_H$	PLL0 / 1	≤ 160 MHz		
0015$_H$	PLL0 / 2	–		
0017$_H$	PLL0 / 4			
001A$_H$	PLL0 / 8			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
003D$_H$	External IISA clock IISAACK			
003E$_H$	External MLB clock MLBA0CLK			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(15) Clock domain ISO1_16

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_116		**Power supply area: Isolated area 1**		
Address	**FF42 A100$_H$**	**Clock domain: ISO1_16**		
Initial value	**0000 0000$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_116	IISABRG20: IISABRG20SCK
0014$_H$	PLL0 / 1	≤ 160 MHz		
0015$_H$	PLL0 / 2	–		
0017$_H$	PLL0 / 4			
001A$_H$	PLL0 / 8			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
003D$_H$	External IISA clock IISAACK			
003E$_H$	External MLB clock MLBA0CLK			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(16) Clock domain ISO1_17

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_117		**Power supply area: Isolated area 1**		
Address	**FF42 A110$_H$**	**Clock domain: ISO1_17**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_117	IISA4: PCLK
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(17) Clock domain ISO1_22

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_122		**Power supply area: Isolated area 1**		
Address	**FF42 A160$_H$**	**Clock domain: ISO1_22**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_122	PCM0: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(18) Clock domain ISO1_23

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_123		**Power supply area: Isolated area 1**		
Address	**FF42 A170$_H$**	**Clock domain: ISO1_23**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_123	PCM1: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

(19) Clock domain ISO1_24

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_124		**Power supply area: Isolated area 1**		
Address	**FF42 A180$_H$**	**Clock domain: ISO1_24**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_124	MLB0: ram_clk/ sys_clk
0014$_H$	PLL0 / 1	≤ 112 MHz		
0015$_H$	PLL0 / 2	≤ 80 MHz		
0017$_H$	PLL0 / 4	–		
001C$_H$	PLL1 / 1	≤ 112 MHz		
001D$_H$	PLL1 / 2	≤ 80 MHz		
001F$_H$	PLL1 / 4	–		
0024$_H$	PLL2 / 1	≤ 112 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
0000H	No clock selected			
Other than the above	Setting prohibited			

(20) Clock domain ISO1_28

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_128		**Power supply area: Isolated area 1**		
Address	**FF42 A1C0$_H$**	**Clock domain: ISO1_28**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_128	DNFAn: DNFATCKI
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
Other than the above	Setting prohibited			

(21) Clock domain ISO1_29

<R> To stop the clock source selected by the CKSC_mn register by using software before the system enters STOP or DEEPSTOP mode, specify “No clock selected” by using the CKSC_mn register in advance. Alternatively set the STPMK_mn bit of the CKSC_mn register to 1.

Clock selector control register: CKSC_129		**Power supply area: Isolated area 1**		
Address	**FF42 A1D0$_H$**	**Clock domain: ISO1_29**		
Initial value	**0000 000E$_H$**			
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: Clock**
0007$_H$	High-speed IntOsc (8 MHz) / 1	–	CKSCLK_129	IISA5: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1	≤ 80 MHz		
001D$_H$	PLL1 / 2	≤ 60 MHz		
001F$_H$	PLL1 / 4	–		
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1	≤ 80 MHz		
0025$_H$	PLL2 / 2	≤ 60 MHz		
0027$_H$	PLL2 / 4	–		
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected			
Other than the above	Setting prohibited			

9.5 Clock Controller Registers

This section describes all registers of the clock controller.

9.5.1 Clock controller register overview

The clock controller is controlled and operated by the following registers:

Table 9-4 Clock controller registers

Register name	Symbol	Address
Clock generator registers:		
MainOsc enable register	MOSCE	FF42 1010_H
MainOsc status register	MOSCS	FF42 1014_H
MainOsc control register	MOSCC	FF42 1018_H
MainOsc stabilization time register	MOSCST	FF42 $101C_H$
SubOsc enable register	SOSCE	FF42 1020_H
SubOsc status register	SOSCS	FF42 1024_H
SubOsc stabilization time register	SOSCST	FF42 $102C_H$
High-speed IntOsc enable register	ROSCE	FF42 1000_H
High-speed IntOsc status register	ROSCS	FF42 1004_H
PLL0 enable register	PLLE0	FF42 5000_H
PLL0 status register	PLLS0	FF42 5004_H
PLL0 control register	PLLC0	FF42 5008_H
PLL0 stabilization time register	PLLST0	FF42 $500C_H$
PLL1 enable register	PLLE1	FF42 5010_H
PLL1 status register	PLLS1	FF42 5014_H
PLL1 control register	PLLC1	FF42 5018_H
PLL1 stabilization time register	PLLST1	FF42 $501C_H$
PLL2 enable register	PLLE2	FF42 5020_H
PLL2 status register	PLLS2	FF42 5024_H
PLL2 control register	PLLC2	FF42 5028_H
PLL2 stabilization time register	PLLST2	FF42 $502C_H$
Clock selector registers:		
Clock selector control/status registers for AWO	CKSC_An CSCSTAT_An	FF42 2020_H to FF42 2094_H
Clock selector control/status registers for Iso0	CKSC_0n CSCSTAT_0n	FF42 6000_H to FF42 6104_H
Clock selector control/status registers for Iso1	CKSC_1n CSCSTAT_1n	FF42 $A010_H$ to FF42 $A1D4_H$

9.5.2 Clock generator registers

(1) MOSCE - MainOsc enable register

This register is used to start and stop the MainOsc and to specify its operation in the standby mode.

This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register PROTCMD2.

For details, see *9.3.6 "Writing to protected registers"*.

Access This register can be read or written in 32-bit units.

Address FF42 1010$_H$

Initial value 0000 0004$_H$. This register is initialized by the power-up reset PURES (a power on clear reset (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products)).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	MOSCSTPMK	MOSCDISTRG	MOSCENTRG
R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	W	W

Table 9-5 MOSCE register contents

Bit position	Bit name	Function
2	MOSC STPMK	MainOsc stop request mask 0: Do not mask stop requests. 1: Mask stop requests. If the MainOsc stop request is masked (the MOSCSTPMK bit = 1), the MainOsc continues operation even in the standby mode. If MOSCSTPMK is 0, the MainOsc is stopped in the standby mode and is restarted upon waking up from the mode.
1	MOSC DISTRG	MainOsc stop trigger 0: No function 1: Stop the MainOsc. Reading this bit always returns 0.
0	MOSC ENTRG	MainOsc enable trigger 0: No function 1: Start the MainOsc. Reading this bit always returns 0.

(2) MOSCS - MainOsc status register

This register provides information about each status of the MainOsc.

Access This register is read-only, in 32-bit units.

Address FF42 1014_H

Initial value $0000\ 0000_H$. This register is initialized by the power-up reset PURES (a power on clear reset (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products)).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	MOSC STP ACK	MOSC CLK EN	MOSC CLK ACT	MOSC CLK STAB
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 9-6 MOSCS register contents

Bit position	Bit name	Function
3	MOSC STPACK	MainOsc standby stop acknowledge status 0: Stop acknowledgment is inactive. 1: Stop acknowledgment is active.
2	MOSC CLKEN	MainOsc enable status 0: The MainOsc is disabled. 1: The MainOsc is enabled.
1	MOSC CLKACT	MainOsc activation status 0: The MainOsc is inactive. 1: The MainOsc is active.
0	MOSC CLKSTAB	MainOsc stabilization status 0: The MainOsc is unstable. 1: The MainOsc is stable.

(3) MOSCC - MainOsc control register

This register is used to start and stop the MainOsc and to specify its operation in the standby mode.

This register can only be written if the MainOsc is disabled (the MOSCS.MOSCCLKEN bit = 0).

Access This register can be read or written in 32-bit units.

Address FF42 1018_H

Initial value $0000\ 0000_H$. This register is initialized by the power-up reset PURES (a power on clear reset (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products)).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SHTSTBY	AMPSEL[1:0]	
R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W

Table 9-7 MOSCC register contents

Bit position	Bit name	Function
2	SHTSTBY	Short stabilization time mode 0: Normal stabilization time mode: Specify the MainOsc amplification gain by using AMPSEL[1:0]. 1: Short stabilization time mode: The MainOsc amplification gain is the maximum.
1, 0	AMPSEL[1:0]	MainOsc frequency selection **AMPSEL[1:0]** / **MainOsc frequency** 00_B / 16 MHz < f_X ≤ 20 MHz 01_B / 8 MHz < f_X ≤ 16 MHz 10_B / 4 MHz < f_X ≤ 8 MHz 11_B / 4 MHz The AMPSEL[1:0] bits specify the MainOsc amplification gain in the normal stabilization time mode (the SHTSTBY bit = 0). In the normal stabilization time mode (SHTSTBY = 1), the AMPSEL[1:0] bits have no effect.

(4) MOSCST - MainOsc stabilization time register

This register is used to specify the MainOsc stabilization time.

This register can be written only when the MainOsc is disabled (the MOSCS.MOSCCLKEN bit = 0).

Access This register can be read or written in 32-bit units.

Address FF42 101C$_H$

Initial value 0000 0000$_H$. This register is initialized by the power-up reset PURES (a power on clear reset (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products)).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	MOST[3:0]			
R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W

Table 9-8 MOSCST register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>3 to 0</td><td>MOST[3:0]</td><td>MainOsc stabilization time setting
By default, the MainOsc stabilization counter is operating with the high-speed IntOsc.
If the high-speed IntOsc is disabled (the ROSCS.ROSCCLKEN bit = 0), the stabilization counter clock is automatically changed to the low-speed IntOsc.
<table>
<tr><th rowspan="2">MOST[3:0]</th><th colspan="2">MainOsc stabilization time</th></tr>
<tr><th>When the high-speed IntOsc is enabled (ROSCS. ROSCCLKEN = 1)</th><th>When the high-speed IntOsc is disabled (ROSCS. ROSCCLKEN = 0)</th></tr>
<tr><td>0000_B</td><td>2^2 / 8 MHz = 0.5 µs</td><td>2^2 / 240 kHz = 16.7 µs</td></tr>
<tr><td>0001_B</td><td>2^3 / 8 MHz = 1 µs</td><td>2^3 / 240 kHz = 33.3 µs</td></tr>
<tr><td>0010_B</td><td>2^4 / 8 MHz = 2 µs</td><td>2^4 / 240 kHz = 66.7 µs</td></tr>
<tr><td>0011_B</td><td>2^5 / 8 MHz = 4 µs</td><td>2^5 / 240 kHz = 133 µs</td></tr>
<tr><td>0100_B</td><td>2^6 / 8 MHz = 8 µs</td><td>2^6 / 240 kHz = 267 µs</td></tr>
<tr><td>0101_B</td><td>2^7 / 8 MHz = 16 µs</td><td>2^7 / 240 kHz = 533 µs</td></tr>
<tr><td>0110_B</td><td>2^8 / 8 MHz = 32 µs</td><td>2^8 / 240 kHz = 1.067 ms</td></tr>
<tr><td>0111_B</td><td>2^9 / 8 MHz = 64 µs</td><td>2^9 / 240 kHz = 2.133 ms</td></tr>
<tr><td>1000_B</td><td>2^{10} / 8 MHz = 128 µs</td><td>2^{10} / 240 kHz = 4.267 ms</td></tr>
<tr><td>1001_B</td><td>2^{11} / 8 MHz = 256 µs</td><td>2^{11} / 240 kHz = 8.533 ms</td></tr>
<tr><td>1010_B</td><td>2^{12} / 8 MHz = 512 µs</td><td>2^{12} / 240 kHz = 17.06 ms</td></tr>
<tr><td>1011_B</td><td>2^{13} / 8 MHz = 1.024 ms</td><td>2^{13} / 240 kHz = 34.13 ms</td></tr>
<tr><td>1100_B</td><td>2^{14} / 8 MHz = 2.048 ms</td><td>2^{14} / 240 kHz = 68.27 ms</td></tr>
<tr><td>1101_B</td><td>2^{15} / 8 MHz = 4.096 ms</td><td>2^{15} / 240 kHz = 136.5 ms</td></tr>
<tr><td>1110_B</td><td>2^{16} / 8 MHz = 8.192 ms</td><td>2^{16} / 240 kHz = 273.1 ms</td></tr>
<tr><td>1111_B</td><td>2^{17} / 8 MHz = 16.38 ms</td><td>2^{17} / 240 kHz = 546.1 ms</td></tr>
</table>
Note
• 8 MHz (Typ.): High-speed IntOsc
• 240 kHz (Typ.): Low-speed IntOsc</td></tr>
</table>

(5) SOSCE - SubOsc enable register

This register is used to start and stop the SubOsc and to specify its operation in the standby mode.

This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register PROTCMD2.

For details, see *9.3.6 "Writing to protected registers"*.

Access This register can be read or written in 32-bit units.

Address FF42 1020_H

Initial value 0000 0004_H. This register is initialized by the power-up reset PURES (a power on clear reset (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products)).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1[a]	SOSC DIS TRG	SOSC EN TRG
R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W

a) Be sure to write 1 to bit 2.

Table 9-9 SOSCE register contents

Bit position	Bit name	Function
1	SOSC DISTRG	SubOsc stop trigger 0: No function 1: Stop the SubOsc. Reading this bit always returns 0.
0	SOSC ENTRG	SubOsc start trigger 0: No function 1: Start the SubOsc. Reading this bit always returns 0.

(6) SOSCS - SubOsc status register

This register provides various types of SubOsc status information.

Access This register is read-only, in 32-bit units.

Address FF42 1024_H

Initial value $0000\ 0000_H$. This register is initialized by the power-up reset PURES (a power on clear reset (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products)).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SOSC STP ACK	SOSC CLK EN	SOSC CLK ACT	SOSC CLK STAB
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 9-10 SOSCS register contents

Bit position	Bit name	Function
3	SOSC STPACK	SubOsc standby stop acknowledge status 0: Stop acknowledgment is inactive. 1: Stop acknowledgment is active.
2	SOSC CLKEN	SubOsc enable status 0: SubOsc is disabled. 1: SubOsc is enabled.
1	SOSC CLKACT	SubOsc activation status 0: SubOsc is inactive. 1: SubOsc is active.
0	SOSC CLKSTAB	SubOsc stabilization status 0: SubOsc is unstable. 1: SubOsc is stable.

(7) SOSCST - SubOsc stabilization time register

This register is used to specify the SubOsc stabilization time.

This register can only be written if the SubOsc is disabled (the SOSCS.SOSCCLKEN bit = 0).

Access This register can be read or written in 32-bit units.

Address FF42 102C$_H$

Initial value 0000 0000$_H$. This register is initialized by the power-up reset PURES (a power on clear reset (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products)).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOST[2:0]		
R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W

Table 9-11 SOSCST register contents

Bit position	Bit name	Function
2 to 0	SOST[2:0]	SubOsc stabilization time setting By default, the SubOsc stabilization counter is operating with the high-speed IntOsc. If the high-speed IntOsc is disabled (the ROSCS.ROSCCLKEN bit = 0), the stabilization counter clock is automatically changed to the low-speed IntOsc.

SOST[2:0]	SubOsc stabilization time	
	When the high-speed IntOsc is enabled (ROSCS. ROSCCLKEN = 1)	When the high-speed IntOsc is disabled (ROSCS. ROSCCLKEN = 0)
000_B	2^{19} / 8 MHz = 65.54 ms	2^{19} / 240 kHz = 2.184 s
001_B	2^{20} / 8 MHz = 131 ms	2^{20} / 240 kHz = 4.369 s
0010_B	2^{21} / 8 MHz = 262 ms	2^{21} / 240 kHz = 8.738 s
011_B	2^{22} / 8 MHz = 524ms	2^{22} / 240 kHz = 17.4 s
100_B	2^{23} / 8 MHz = 1.048 s	2^{23} / 240 kHz = 34.9 s
101_B	2^{24} / 8 MHz = 2.096 s	2^{24} / 240 kHz = 69.9 s
110_B	2^{25} / 8 MHz = 4.192 s	2^{25} / 240 kHz = 139 s
111_B	2^{26} / 8 MHz = 8.384 s	2^{26} / 240 kHz = 279 s

Note
- 8 MHz (Typ.): High-speed IntOsc
- 240 kHz (Typ.): Low-speed IntOsc

(8) ROSCE - High-speed IntOsc enable register

This register is used to start and stop the high-speed IntOsc and to specify its operation in the standby mode.

This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register PROTCMD2.

For details, see *9.3.6 "Writing to protected registers"*.

Access This register can be read or written in 32-bit units.

Address FF42 1000_H

Initial value 0000 0004_H. This register is initialized by the power-up reset PURES (a power on clear reset (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products)).

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<R>	0	0	0	0	0	0	0	0	0	0	0	0	0	ROSC STP MK	0[a]	ROSC EN TRG
<R>	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R	R/W

<R> a) Be sure to clear bit 1 to 0.

<R> **Table 9-12 ROSCE register contents**

Bit position	Bit name	Function
2	ROSC STPMK	High-speed IntOsc stop request mask 0: Do not mask stop requests. 1: Mask stop requests. If the high-speed IntOsc stop request is masked (the ROSCSTPMK bit = 1), the high-speed IntOsc continues operation even in the standby mode. If the ROSCSTPMK bit is 0, the high-speed IntOsc is stopped in the standby mode and is restarted upon waking up from the mode.
0	ROSC ENTRG	High-speed IntOsc start trigger 0: No function 1: Start the high-speed IntOsc. Reading this bit always returns 0.

(9) ROSCS - high-speed IntOsc status register

This register provides various types of high-speed IntOsc status information.

Access This register is read-only, in 32-bit units.

Address FF42 1004_H

Initial value 0000 0007_H. This register is initialized by the power-up reset PURES (a power on clear reset (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products)).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	ROSC STP ACK	ROSC CLK EN	ROSC CLK ACT	ROSC CLK STAB
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 9-13 ROSCS register contents

Bit position	Bit name	Function
3	ROSC STPACK	High-speed IntOsc standby stop acknowledge status 0: Stop acknowledgment is inactive. 1: Stop acknowledgment is active.
2	ROSC CLKEN	High-speed IntOsc enable status 0: The high-speed IntOsc is disabled. 1: The high-speed IntOsc is enabled.
1	ROSC CLKACT	High-speed IntOsc activation status 0: The high-speed IntOsc is inactive. 1: The high-speed IntOsc is active.
0	ROSC CLKSTAB	High-speed IntOsc stabilization status 0: The high-speed IntOsc is unstable. 1: The high-speed IntOsc is stable.

(10) PLLEk - PLLk enable register

This register is used to start and stop the PLLk and to specify its operation in the standby mode.

This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register PROTCMD2.

For details, see *9.3.6 "Writing to protected registers"*.

Access This register can be read or written in 32-bit units.

Address PLLE0: FF42 5000$_H$, PLLE1: FF42 5010$_H$, PLLE2: FF42 5020$_H$

Initial value 0000 0004$_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	PLLkSTPMK	PLLkDISTRG	PLLkENTRG
R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W

Table 9-14 PLLEk register contents

Bit position	Bit name	Function
2	PLLkSTPMK	PLLk stop request mask 0: Do not mask stop requests. 1: Mask stop requests. If the PLLk stop request is masked (the PLLkSTPMK bit = 1), the PLLk continues operation even in the standby mode. If PLLkSTPMK = 0, the PLLk is stopped in the standby mode and is restarted upon waking up from the mode.
1	PLLkDISTRG	PLLk stop trigger 0: No function 1: Stop PLLk. Reading this bit always returns 0.
0	PLLkENTRG	PLLk start trigger 0: No function 1: Start PLLk. Reading this bit always returns 0.

(11) PLLSk - PLLk status register

This register provides various types of PLLk status information.

Access This register is read-only, in 32-bit units.

Address PLLS0: FF42 5004_H, PLLS1: FF42 5014_H, PLLS2: FF42 5024_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	PLLkSTPACK	PLLkCLKEN	PLLkCLKACT	PLLkCLKSTAB
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 9-15 PLLSk register contents

Bit position	Bit name	Function
3	PLLkSTPACK	PLLk standby stop acknowledge status 0: Stop acknowledgment is inactive. 1: Stop acknowledgment is active.
2	PLLkCLKEN	PLLk enable status 0: PLLk is disabled. 1: PLLk is enabled.
1	PLLkCLKACT	PLLk activation status 0: PLLk is inactive. 1: PLLk is active.
0	PLLkCLKSTAB	PLLk stabilization status 0: PLLk is unstable. 1: PLLk is stable.

(12) PLLCk - PLLk control register

This register is used to specify the PLLk output clock f_{PLk} frequency.

This register can only be written if PLLk is disabled (the PLLSk.PLLkCLKEN bit = 0).

Access This register can be read or written in 32-bit units.

Address PLLC0: FF42 5008_H, PLLC1: FF42 5018_H, PLLC2: FF42 5028_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Note that, after the initialization operation, specifying $0000\ 0000_H$ is prohibited when setting up the register again.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0[a]	MS	PC[1:0]		ADJ[2:0]			MD[1:0]	
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

a) Be sure to clear bit 24 to “0”.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S[1:0]		M[3:0]				P[1:0]		0	N[6:0]						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-16 PLLCk register contents (1/2)

Bit position	Bit name	Function
23	MS	PLL mode selection 0: SSCG mode (modulation mode) 1: PLL mode
22, 21	PC[1:0]	Modulation mode selection **PC[1:0]** / **Modulation mode** $0X_B$ / Fixed frequency 10_B / Down spread modulation 11_B / Center spread modulation
20 to 18	ADJ[2:0]	Frequency modulation range selection **ADJ[1:0]** / **Frequency modulation range** 000_B / Setting prohibited 001_B / 1% 010_B / 2% 011_B / 3% 100_B / 4% 101_B / 5% 110_B / Setting prohibited 111_B / Setting prohibited

Table 9-16 PLLCk register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>17, 16</td><td>MDL[1:0]</td><td>Frequency modulation cycle control
<table>
<tr><th>MDL[1:0]</th><th>Modulation frequency</th></tr>
<tr><td>00_B</td><td>40 kHz</td></tr>
<tr><td>01_B</td><td>50 kHz</td></tr>
<tr><td>10_B</td><td>60 kHz</td></tr>
<tr><td>11_B</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>15, 14</td><td>S[1:0]</td><td>VCO input frequency division in the SSCG mode
<table>
<tr><th>S[1:0]</th><th>Input frequency range</th></tr>
<tr><td>00_B</td><td>1.0 MHz ≤ f_{VCOIN} < 1.2 MHz</td></tr>
<tr><td>01_B</td><td>1.2 MHz ≤ f_{VCOIN} < 1.4 MHz</td></tr>
<tr><td>10_B</td><td>1.4 MHz ≤ f_{VCOIN} < 1.7 MHz</td></tr>
<tr><td>11_B</td><td>1.7 MHz ≤ f_{VCOIN} < 2.0 MHz</td></tr>
</table></td></tr>
<tr><td>13 to 10</td><td>M[3:0]</td><td>Mr value specification
SSCG mode: Mr = M[3:0] + 1
PLL mode M[3:0] = 0: Mr = 1</td></tr>
<tr><td>9, 8</td><td>P[1:0]</td><td>P divider selection
<table>
<tr><th rowspan="2">P[1:0]</th><th rowspan="2">Pr</th><th colspan="2">PLL output frequency range</th></tr>
<tr><th>SSCG mode</th><th>PLL mode</th></tr>
<tr><td>00_B</td><td>0.5</td><td>Setting prohibited</td><td>Setting prohibited</td></tr>
<tr><td>01_B</td><td>1</td><td>100 MHz to 160 MHz</td><td>100 MHz to 160 MHz</td></tr>
<tr><td>10_B</td><td>2</td><td>50 MHz to 100 MHz</td><td>50 MHz to 120 MHz</td></tr>
<tr><td>11_B</td><td>4</td><td>25 MHz to 50 MHz</td><td>25 MHz to 60 MHz</td></tr>
</table></td></tr>
<tr><td>6 to 0</td><td>N[6:0]</td><td>Nr value specification
<table>
<tr><th rowspan="2">N[6:0]</th><th colspan="2">Nr value</th></tr>
<tr><th>SSCG mode</th><th>PLL mode</th></tr>
<tr><td>$000\ 0100_B$</td><td>–</td><td>5</td></tr>
<tr><td>$000\ 0101_B$</td><td>–</td><td>6</td></tr>
<tr><td>...</td><td>...</td><td>...</td></tr>
<tr><td>$010\ 0111_B$</td><td>–</td><td>40</td></tr>
<tr><td>$011\ 0001_B$</td><td>50</td><td>–</td></tr>
<tr><td>$011\ 0010_B$</td><td>51</td><td>–</td></tr>
<tr><td>...</td><td>...</td><td>...</td></tr>
<tr><td>$110\ 0010_B$</td><td>99</td><td>–</td></tr>
<tr><td>$110\ 0011_B$</td><td>100</td><td>–</td></tr>
</table></td></tr>
</table>

(13) PLLSTk - PLLk stabilization time register

This register is used to specify the PLLk stabilization time.

This register can only be written if PLLk is disabled (the PLLSk.PLLkCLKEN bit = 0).

Access This register can be read or written in 32-bit units.

Address PLLST0: FF42 500C$_H$, PLLST1: FF42 501C$_H$, PLLST2: FF42 502C$_H$

Initial value 0000 0000$_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	PLLSTk[2:0]		
R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W

Table 9-17 PLLSTk register contents

Bit position	Bit name	Function
2 to 0	PLLSTk[2:0]	PLLk stabilization time setting By default, the PLLk stabilization counter is operating with the high-speed IntOsc. If the high-speed IntOsc is disabled (the ROSCS.ROSCCLKEN bit = 0), the stabilization counter clock is automatically changed to the low-speed IntOsc.

PLLSTk[2:0]	PLLk stabilization time	
	When the high-speed IntOsc is enabled (ROSCS. ROSCCLKEN = 1)	When the high-speed IntOsc is disabled (ROSCS. ROSCCLKEN = 0)
000_B	2^7 / 8 MHz = 16 µs	2^7 / 240 kHz = 533 µs
001_B	2^8 / 8 MHz = 32 µs	2^8 / 240 kHz = 1.067 ms
010_B	2^9 / 8 MHz = 64 µs	2^9 / 240 kHz = 2.133 ms
011_B	2^{10} / 8 MHz = 128 µs	2^{10} / 240 kHz = 4.267 ms
100_B	2^{11} / 8 MHz = 256 µs	2^{11} / 240 kHz = 8.533 ms
101_B	2^{12} / 8 MHz = 512 µs	2^{12} / 240 kHz = 17.057 ms
110_B	2^{13} / 8 MHz = 1.024 ms	2^{13} / 240 kHz = 34.133 ms
111_B	2^{14} / 8 MHz = 2.048 ms	2^{14} / 240 kHz = 68.267 ms

Note
- 8 MHz (Typ.): High-speed IntOsc
- 240 kHz (Typ.): Low-speed IntOsc

9.5.3 Protection command register details

(1) PROTCMDm – Protection command register m (m = 0 to 3)

The protection command register is used to start the write protection cancellation sequence for a write protected register.

Access This register is write-only, in 8-bit units.

Reading on this register returns always 0.

Address PROTCMD0: FF42 4000$_H$
PROTCMD1: FF42 8000$_H$
PROTCMD2: FF42 0300$_H$
PROTCMD3: FF42 0308$_H$

Initial value 00$_H$. This register is initialized by any reset.

7	6	5	4	3	2	1	0
PCMD7	PCMD6	PCMD5	PCMD4	PCMD3	PCMD2	PCMD1	PCMD0
W	W	W	W	W	W	W	W

For details, see *9.3.6 “Writing to protected registers”*.

Table 9-18 PROTCMDm register contents

Bit position	Bit name	Function
7 to 0	PCM[7:0]	Protection commands to enable writing to Isolated area m registers

(2) PROTSm – Protection status register m (m = 0 to 3)

This register shows the status of the protection cancellation sequence operated by way of PROTCMDm.

Access This register is read-only, in 8-bit units.

Writing to this register is ignored.

Address PROTS0: FF42 4004$_H$
PROTS1: FF42 8004$_H$
PROTS2: FF42 0304$_H$
PROTS3: FF42 030C$_H$

Initial value 00$_H$. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PROTERR
R	R	R	R	R	R	R	R

Table 9-19 PROTSm register contents

Bit position	Bit name	Function
0	PROTERR	Protected write sequence error monitor 0: No protection error occurred. 1: A protection error occurred.

9.5.4 Clock selector control register

(1) CKSC_mn – Clock selector control registers

These registers select the clock for all clock domains.

This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register for each power supply area.

m = 0: Iso0 Writing to CKSC_0n registers is protected and can only be written by using the protection register PRTCMD0.

m = 1: Iso1 Writing to CKSC_1n registers is protected and can only be written by using the protection register PRTCMD1.

m = A: AWO Writing to CKSC_An registers is protected and can only be written by using the protection register PRTCMD2.

For details, see *9.3.6 "Writing to protected registers"*.

Access These registers can be read or written in 32-bit units.

Address Isolated area 0: FF42 6000_H + n x 16
Isolated area 1: FF42 $A000_H$ + n x 16
Always-On area: FF42 2000_H + n x 16

Initial value See *9.4 "Clock Selection"*. This register is initialized by any reset.

31	30	29	28	27	26	25	24
CKSCID_mn[30:23]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
CKSCID_mn[22:15]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
CKSCID_mn[14:7]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
CKSCID_mn[6:0]							STPMK_mn
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-20 CKSC_mn register contents

Bit position	Bit name	Function
31 to 1	CKSCID_mn[30:0]	Clock source ID These bits specify the clock CKSCLK_mn for clock domain mn[30:0].
0	STPMK_mn	Controls the clock output CKSCLK_mn during the standby mode: 0: Stop outputting CKSCLK_mn. 1: Continue outputting CKSCLK_mn.

(2) CSCSTAT_mn – Clock selector status registers

This register indicates the clock source ID currently selected by CKSC_mn and whether the clock is enabled.

Access This register can be read or written in 32-bit units.

Address Isolated area 0: FF42 6004_H + n x 16
Isolated area 1: FF42 $A004_H$ + n x 16
Always-On area: FF42 2004_H + n x 16

Initial value See *9.4 "Clock Selection"*. This register is initialized by any reset.

31	30	29	28	27	26	25	24
CLKSELID_mn[30:23]							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
CLKSELID_mn[22:15]							
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
CLKSELID_mn[14:7]							
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
CLKSELID_mn[6:0]							CLKACT_mn
R	R	R	R	R	R	R	R

Table 9-21 CSCSTAT_mn register contents

Bit position	Bit name	Function
31 to 1	CLKSELID_mn[30:0]	Clock source ID of the clock currently selected by CKSC_mn
0	CLKACT_mn	Indicates whether CKSCLK_nm is enabled or disabled: 0: CKSCLK_nm is disabled. 1: CKSCLK_nm is enabled.

9.6 Clock Monitor A (CLMA)

This section contains a generic description of clock monitor A (CLMA).

The first section describes all properties specific to the V850E2/Sx4-H, such as instances, register base addresses, input/output signal names, etc.

The subsequent sections describe the features that apply to all clock controller implementations.

9.6.1 V850E2/Sx4-H CLMA features

Number of channels

This microcontroller has the following number of channels of CLMA:

Table 9-22 Channel

CLMA	
Number of channels	3
Name	CLMA0, CLMA2, CLMA3

Instances index n

In this chapter, each channel of clock monitor A is identified using n (where n = 0, 2, or 3). For example, descriptions such as control register 0 (CLMAnCTL0) or CLMAn are used.

Register addresses

All CLMAn register addresses are given as address offsets from the individual base address <CLMAn_base>.
The base address <CLMAn_base> of each CLMAn is listed in the following table:

Table 9-23 Register base address <CLMAn_base>

CLMAn	<CLMAn_base> address
CLMA0	FF80 2000_H
CLMA2	FF80 4000_H
CLMA3	FF80 5000_H

Clock supply

All monitor and sampling clocks of clock monitor A are listed in the following table:

Table 9-24 CLMAn clock supply

CLMAn clock	Function	Connected to:
CLMA0:		
CLMATSMP	CLMA0 sampling clock	Low-speed IntOsc (240 kHz)
CLMATMON	CLMA0 monitored clock	MainOsc
PCLK	PBUS clock	CKSCLK_A02
CLMA2:		
CLMATSMP	CLMA2 sampling clock	Low-speed IntOsc (240 kHz)
CLMATMON	CLMA2 monitored clock	High-speed IntOsc (8 MHz)
PCLK	PBUS clock	CKSCLK_A02
CLMA3:		
CLMATSMP	CLMA3 sampling clock	High-speed IntOsc (8 MHz)
CLMATMON	CLMA3 monitored clock	PLL0
PCLK	PBUS clock	CKSCLK_005

Interrupts and reset outputs

The interrupts and reset outputs of CLMAn are listed in the following table:

Table 9-25 CLMA interrupts and reset outputs

CLMAn signal	Function	Connected to:
CLMA0:		
CLMARES	CLMA0 error reset	Reset controller $\overline{\text{CLMA0RES}}$
CLMATI	CLMA0 error interrupt request	Not connected
CLMA2:		
CLMARES	CLMA2 error reset	Reset controller $\overline{\text{CLMA2RES}}$
CLMATI	CLMA2 error interrupt request	Not connected
CLMA3:		
CLMARES	CLMA3 error reset	Reset controller $\overline{\text{CLMA3RES}}$
CLMATI	CLMA3 error interrupt request	Not connected

<R>

9.6.2 CLMA enable

Clock monitoring by the clock monitors starts once the clock to be monitored is has stabilized.

- CLMA0 automatically starts once the MainOsc clock f_X has stabilized.
- CLMA2 automatically starts once the high-speed IntOsc clock f_{RH} has stabilized.
- CLMA0 automatically starts once the PLL0 clock f_{PL0} has stabilized.

If the monitor clock stops in the STOP standby mode, the corresponding clock monitor is disabled. Later, when the monitor clock starts oscillating and has stabilized, clock monitor operation resumes.

9.6.3 Functional overview

The clock monitor CLMAn indicates an abnormal frequency of the monitored clock.

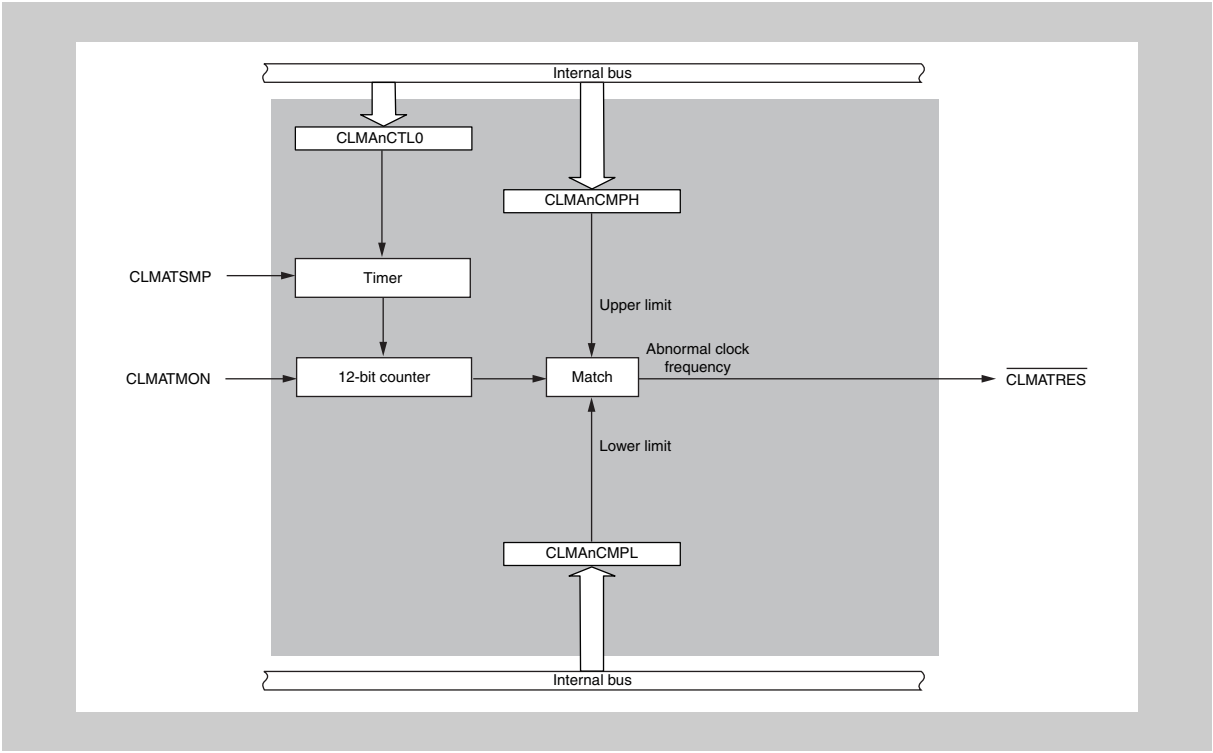
Overview This clock monitor has the following features:

- Continuous monitoring of the frequency of the input clock CLMATMON by using the sampling clock CLMATSMP
- Outputs a reset request signal if the clock frequency is abnormal

<R>

Note Once enabled, CLMAn can only be disabled by a reset.

The following figure shows the main components of the clock monitor.



<R> **Figure 9-9 Block diagram of clock monitor A**

<R>

9.6.4 Description

The clock monitor CLMAn is used to ensure that the frequency of a clock (CLMATMON) stays between certain limits.

(1) Detection of abnormal clock frequencies

Detection method

1. CLMAn counts the rising edges of the monitored clock CLMATMON within 16 cycles of the sampling clock CLMATSMP and then compares the counter with the specified thresholds:
   – CLMAnCMPL.CLMAnCMPL[11:0] defines the lower threshold.
   – CLMAnCMPH.CLMAnCMPH[11:0] defines the upper threshold.
2. When CLMATMON stops or its frequency is too low, the counter falls below CLMAnCMPL.CLMAnCMPL[11:0].
3. When the frequency of CLMATMON is too high, the counter exceeds CLMAnCMPH.CLMAnCMPH[11:0].

In both cases, CLMAn indicates an abnormal clock frequency as described in *(2) "Indication of abnormal clock frequency" on page 495*.

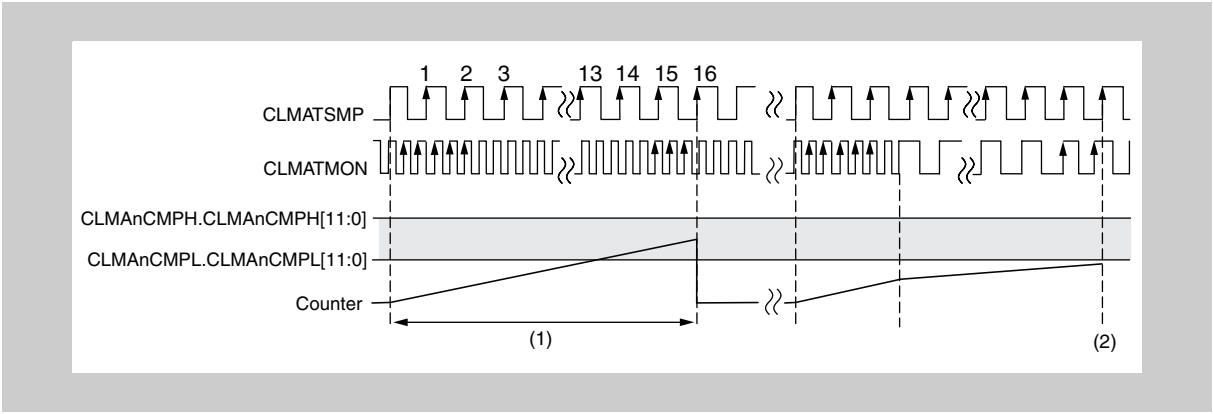


Figure 9-10 Example: When $f_{CLMATMON}$ is low

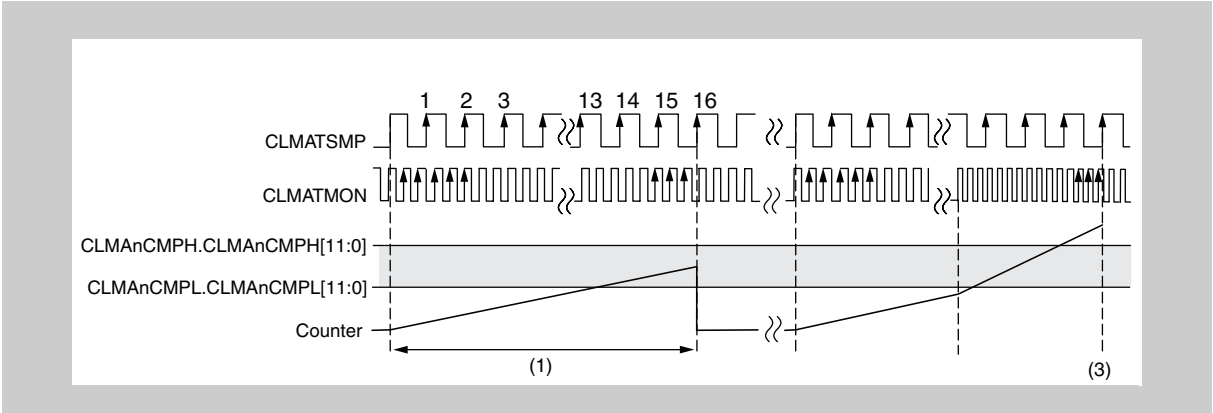


Figure 9-11 Example: When $f_{CLMATMON}$ is high

Note When $f_{CLMATMON}$ changes within the sampling interval, the counter might be within the valid range although $f_{CLMATMON}$ became too high/low.

The abnormal $f_{CLMATMON}$ is detected one sampling interval later.

(a) Calculation of thresholds CLMAnCMPL.CLMAnCMPL[11:0] and CLMAnCMPH.CLMAnCMPH[11:0]

The compare registers CLMAnCMPL and CLMAnCMPH are set up using the minimum and maximum number of clock cycles of CLMATMON that are assumed to be valid within 16 cycles of the sampling clock CLMATSMP.

The expected number of clock cycles is denoted by N.

$$\frac{8}{f_{CLMATSMP}} = \frac{N}{f_{CLMATMON}}$$

$$N = \frac{f_{CLMATMON}}{f_{CLMATSMP}} \times 16$$

Considering the allowed frequency deviations of CLMATMON and CLMATSMP, the threshold values can be calculated using the following formulas:

$$\text{Lower threshold} = N_{min} = \frac{f_{CLMATMON(min)}}{f_{CLMATSMP(max)}} \times 16$$

$$\text{Upper threshold} = N_{max} = \frac{f_{CLMATMON(max)}}{f_{CLMATSMP(min)}} \times 16$$

Example If $f_{CLMATSMP}$ = 240 kHz (± 8%) and $f_{CLMATMON}$ = 16 MHz (± 5%), the recommended threshold values are the following:

$$N_{min} = 15{,}200 / 259.2 \times 16$$
$$= 938.27$$
$$\text{CLMAnCMPL} = 938 = 03AA_H$$

$$N_{max} = 16{,}800 / 220.8 \times 16$$
$$= 1217.38$$
$$\text{CLMAnCMPH} = 1218 = 04C2_H$$

Minimum thresholds The following restrictions must be taken into account:

- CLMAnCMPL ≥ 0001_H
- CLMAnCMPH ≥ CLMAnCMPL + 0003_H

(2) Indication of abnormal clock frequency

If $f_{CLMATMON}$ exceeds the upper threshold value or falls below the lower threshold value, CLMAn operates as follows:

1. The reset request signal $\overline{\text{CLMATRES}}$ is set to the low level.
2. As a result, a system reset $\overline{\text{PRESET}}$ is generated and resets CLMAn.

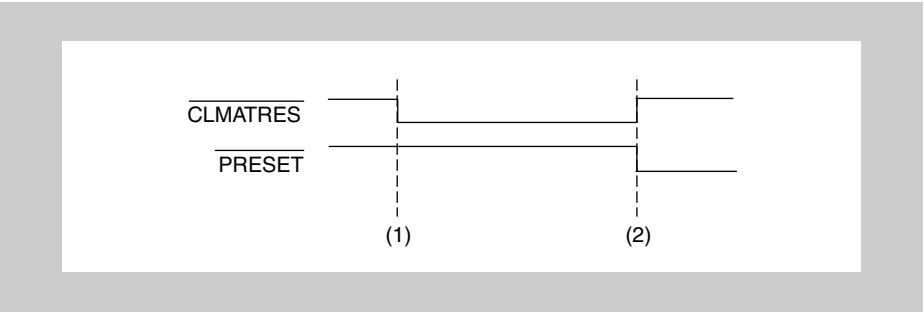


Figure 9-12 Error request signal when $f_{CLMATMON}$ is abnormal

(3) Enabling CLMAn (writing to CLMAnCTL0)

The control register CLMAnCTL0 is a write protected register used to enable CLMAn.

Note CLMAn can only be disabled by a reset, not by writing to CLMAnCTL0.

(a) Write procedure for enabling CLMAn

To set CLMAnCTL0 to 01_H, perform the following sequence of instructions:

1. Write $A5_H$ to CLMAnPCMD.
2. Write CLMAnCTL0 by performing the following sequence:
   - Write 01_H to enable CLMAn.
   - Write the inverted value FE_H.
   - Again, write the intended value 01_H.
3. Read CLMAnCTL0.

   If CLMAnCTL0 is 01_H, CLMAn was enabled successfully.
   Otherwise, check the CLMAnCTL0 write operation status register CLMAnPS:
   - If CLMAnPS is 01_H, the sequence of instructions was executed incorrectly. Restart the sequence from step 1 to retry enabling CLMAn.
   - If CLMAnPS is 00_H, first write 00_H to CLMAnPCMD, and then restart the sequence from step 1 to retry enabling CLMAn.

9.6.5 Clock monitor registers

The clock monitor is controlled and operated by the following registers:

Table 9-26 Clock monitor registers

Register name	Symbol	Address
CLMAn control register 0	CLMAnCTL0	<CLMAn_base> + 00_H
CLMAn comparison register L	CLMAnCMPL	<CLMAn_base> + 08_H
CLMAn comparison register H	CLMAnCMPH	<CLMAn_base> + $0C_H$
CLMAn emulation register 0	CLMAnEMU0	<CLMAn_base> + 18_H

<CLMAn_base> The base addresses <CLMAn_base> of CLMAn is defined in the first part of *"Clock Monitor A (CLMA)"* under the keyword "Register addresses".

(1) CLMAnCTL0 – CLMAn control register 0

This register is used to enable the clock monitor CLMAn.

Access This register can be read or written in 8-bit or 1-bit units.

This register is write-protected, and can only be written by using a special instruction sequence. For details, see *(3) "Enabling CLMAn (writing to CLMAnCTL0)" on page 495*.

Address <CLMAn_base> + 00_H

Initial value This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	CLMAn CLME
R	R	R	R	R	R	R	R/W

Table 9-27 CLMAnCTL0 register contents

Bit position	Bit name	Function
0	CLMAnCLME	Enables/disables the clock monitor: 0: Disable CLMAn. 1: Enable CLMAn.

Caution Write 0 to bits 7 to 1.

(2) CLMAnCMPL – CLMAn comparison register L

This register is used to specify the lower frequency limit.

For details, see *(a) Calculation of thresholds CLMAnCMPL.CLMAnCMPL[11:0] and CLMAnCMPH.CLMAnCMPH[11:0] on page 494*.

Access This register can be read or written in 16-bit units. It can only be written when CLMAn is disabled (CLMAnCTL0.CLMAnCLME = 0).

Address <CLMAn_base> + 08_H

Initial value 0001_H. This register is initialized by any reset.

<R>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CLMAnCMPL[11:0]											
R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-28 CLMAnCMPL register contents

Bit position	Bit name	Function
11 to 0	CLMAnCMPL[11:0]	Specifies the lower threshold • The recommended value is ($f_{CLMATMON(min)}$ x 16) / $f_{CLMATSMP(max)}$. • The minimum value is 0001_H.

<R>

<R> **Caution** Write 0 to bits 15 to 12.

(3) CLMAnCMPH – CLMAn comparison register H

This register is used to specify the upper frequency limit.

For details, see *(a) Calculation of thresholds CLMAnCMPL.CLMAnCMPL[11:0] and CLMAnCMPH.CLMAnCMPH[11:0] on page 494*.

Access This register can be read or written in 16-bit units. It can be written only when CLMAn is disabled (CLMAnCTL0.CLMAnCLME = 0).

Address <CLMAn_base> + $0C_H$

Initial value $03FF_H$. This register is initialized by any reset.

<R>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CLMAnCMPH[11:0]											
R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-29 CLMAnCMPH register contents

<R>

Bit position	Bit name	Function
11 to 0	CLMAnCMPH[11:0]	Specifies the upper threshold. • The recommended value is ($f_{CLMATMON(max)}$ x 16) / $f_{CLMATSMP(min)}$. • The minimum value is CLMAnCMPL + 0003_H.

<R> **Caution** Write 0 to bits 15 to 12.

<R>

(4) CLMAnEMU0 – CLMAn emulation register 0

This register provides bits for emulating frequency deviation errors while the microcontroller is set to the break mode during debugging.

Access This register can be read or written in 8-bit units.

Address <CLMAn_base> + 18_H

Initial value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	CLMAn SLFST	CLMAn SLSLW
R	R	R	R	R	R	R/W	R/W

Table 9-30 CLMAnEMU0 register contents

Bit position	Bit name	Function
0	CLMAnSLFST	Specifies whether $f_{CLMATMON}$ is assumed to be too high during emulation: 0: CLMATMON is within the normal frequency range. 1: CLMATMON is too fast.
1	CLMAnSLSLW	Specifies whether $f_{CLMATMON}$ is assumed to be too low during emulation: 0: CLMATMON is within the normal frequency range. 1: CLMATMON is too slow.

<R> **Cautions**

1. It is prohibited to emulate a CLMATMON value that is too low and another that is too high at the same time. Therefore, CLMAnEMU0 must not be set to 03_H.

<R> 2. Write 0 to bits 7 to 2.

Chapter 10 Standby Controller (STBC)

This chapter describes the standby controller (STBC).

The first section describes all V850E2/Sx4-H specific properties, such as register base addresses and wake-up factors.
The subsequent sections describe the features that apply to all implementations.

10.1 V850E2/Sx4-H STBC Features

STBC reset

The STBC and its registers are initialized by the following reset signal:

Table 10-1 STBC reset signal

<R>

STBC	Reset signal
STBC	System reset SYSRES Power-up reset PURES

Wake-up factors

The wake-up events for terminating a power save mode are controlled and monitored by the following standby controller registers:

- WUFLm, WUFMSKLm, WUFCLm
- WUFMm, WUFMSKMm, WUFCMm
- WUFHm, WUFMSKHm, WUFCHm

The index *m = 0 or 1* denotes the Isolated area 0 or Isolated area 1.

The assignment of the wake-up sources to the control and status register bits is given in the table below.

For details about the wake-up control and status registers, see *10.3.3 “Wake-up event controller register details”*.

Table 10-2 Wake-up factor register assignments (WUFLm, WUFMSKLm, and WUFCLm; m = 0 or 1)

	Assignment of WUFLm, WUFMSKLm, and WUFCLm register bits						Wake-up factors			
	Isolated area 0 wake-up			Isolated area 1 wake-up			Wake-up factors	Module	Power domain	Clock domain
<R>	WUFL000	WUFMSKL000	WUFCL000	WUFL100	WUFMSKL100	WUFCL100	NMI	Port	–	–
	WUFL001	WUFMSKL001	WUFCL001	WUFL101	WUFMSKL101	WUFCL101	INTWDTA0	WDTA0	AWO	CKSCLK_A07
<R>	WUFL002	WUFMSKL002	WUFCL002	WUFL102	WUFMSKL102	WUFCL102	INTLVI	LVI	–	–
	WUFL003	WUFMSKL003	WUFCL003	WUFL103	WUFMSKL103	WUFCL103	INTKR0	KR0	AWO	CKSCLK_A02
	WUFL004	WUFMSKL004	WUFCL004	WUFL104	WUFMSKL104	WUFCL104	INTRTCA0AL	RTCA0		CKSCLK_A09
	WUFL005	WUFMSKL005	WUFCL005	WUFL105	WUFMSKL105	WUFCL105	INTRTCA0R			
	WUFL006	WUFMSKL006	WUFCL006	WUFL106	WUFMSKL106	WUFCL106	INTRTCA01S			
<R>	WUFL007	WUFMSKL007	WUFCL007	WUFL107	WUFMSKL107	WUFCL107	INTP0	Port	–	–
<R>	WUFL008	WUFMSKL008	WUFCL008	WUFL108	WUFMSKL108	WUFCL108	INTP1			
<R>	WUFL009	WUFMSKL009	WUFCL009	WUFL109	WUFMSKL109	WUFCL109	INTP2			
<R>	WUFL010	WUFMSKL010	WUFCL010	WUFL110	WUFMSKL110	WUFCL110	INTP3			
<R>	WUFL011	WUFMSKL011	WUFCL011	WUFL111	WUFMSKL111	WUFCL111	INTP4			
<R>	WUFL012	WUFMSKL012	WUFCL012	WUFL112	WUFMSKL112	WUFCL112	INTP5			
<R>	WUFL013	WUFMSKL013	WUFCL013	WUFL113	WUFMSKL113	WUFCL113	INTP6			
<R>	WUFL014	WUFMSKL014	WUFCL014	WUFL114	WUFMSKL114	WUFCL114	INTP7			
<R>	WUFL015	WUFMSKL015	WUFCL015	WUFL115	WUFMSKL115	WUFCL115	INTP8			
<R>	WUFL016	WUFMSKL016	WUFCL016	WUFL116	WUFMSKL116	WUFCL116	INTP9			
<R>	WUFL017	WUFMSKL017	WUFCL017	WUFL117	WUFMSKL117	WUFCL117	INTP10			
<R>	WUFL018	WUFMSKL018	WUFCL018	WUFL118	WUFMSKL118	WUFCL118	INTP11			
<R>	WUFL019	WUFMSKL019	WUFCL019	WUFL119	WUFMSKL119	WUFCL119	INTP12			
<R>	WUFL020	WUFMSKL020	WUFCL020	WUFL120	WUFMSKL120	WUFCL120	INTP13			
<R>	WUFL021	WUFMSKL021	WUFCL021	WUFL121	WUFMSKL121	WUFCL121	INTP14			
<R>	WUFL022	WUFMSKL022	WUFCL022	WUFL122	WUFMSKL122	WUFCL122	INTP15			
<R>	WUFL023	WUFMSKL023	WUFCL023	WUFL123	WUFMSKL123	WUFCL123	FCN0RX[a]			
<R>	WUFL024	WUFMSKL024	WUFCL024	WUFL124	WUFMSKL124	WUFCL124	FCN1RX[a]			

a) Though the CAN controller FCNn is assigned to Isolated area 1, the CAN bus reception input signal FCNnRX can be selected as a wake-up source in any DEEPSTOP mode.

Table 10-3 Wake-up factor register assignments (WUFMm, WUFMSKMm, and WUFCMm; m = 0 or 1)

Assignment of WUFMm, WUFMSKMm, and WUFCMm register bits						Wake-up factors			
Isolated area 0 wake-up			Isolated area 1 wake-up			Wake-up factors	Module	Power domain	Clock domain
WUFM002	WUFMSKM002	WUFCM002	WUFM102	WUFMSKM102	WUFCM102	INTTAUJ0I0	TAUJ0	AWO	CKSCLK_A03
WUFM003	WUFMSKM003	WUFCM003	WUFM103	WUFMSKM103	WUFCM103	INTTAUJ0I1			
WUFM004	WUFMSKM004	WUFCM004	WUFM104	WUFMSKM104	WUFCM104	INTTAUJ0I2			
WUFM005	WUFMSKM005	WUFCM005	WUFM105	WUFMSKM105	WUFCM105	INTTAUJ0I3			
WUFM010	WUFMSKM010	WUFCM010	WUFM110	WUFMSKM110	WUFCM110	INTADCA0ERR	ADCA0	ISO0	CKSCLK_012
WUFM011	WUFMSKM011	WUFCM011	WUFM111	WUFMSKM111	WUFCM111	INTADCA0I0			
WUFM012	WUFMSKM012	WUFCM012	WUFM112	WUFMSKM112	WUFCM112	INTADCA0I1			
WUFM013	WUFMSKM013	WUFCM013	WUFM113	WUFMSKM113	WUFCM113	INTADCA0I2			
WUFM014	WUFMSKM014	WUFCM014	WUFM114	WUFMSKM114	WUFCM114	INTADCA0LLT			
WUFM015	WUFMSKM015	WUFCM015	WUFM115	WUFMSKM115	WUFCM115	INTWDTA1	WDTA1		CKSCLK_007
WUFM017	WUFMSKM017	WUFCM017	WUFM117	WUFMSKM117	WUFCM117	INTLMA10IT	LMA10		CKSCLK_011
WUFM018	WUFMSKM018	WUFCM018	WUFM118	WUFMSKM118	WUFCM118	INTLMA10IR			
WUFM019	WUFMSKM019	WUFCM019	WUFM119	WUFMSKM119	WUFCM119	INTLMA10IS			
WUFM023	WUFMSKM023	WUFCM023	WUFM123	WUFMSKM123	WUFCM123	INTCSIG4IC	CSIG4		CKSCLK_011
WUFM024	WUFMSKM024	WUFCM024	WUFM124	WUFMSKM124	WUFCM124	INTCSIG4IR			
WUFM025	WUFMSKM025	WUFCM025	WUFM125	WUFMSKM125	WUFCM125	INTCSIG4IRE			
WUFM029	WUFMSKM029	WUFCM029	WUFM129	WUFMSKM129	WUFCM129	INTTAUA0I0	TAUA0		CKSCLK_006
WUFM030	WUFMSKM030	WUFCM030	WUFM130	WUFMSKM130	WUFCM130	INTTAUA0I1			
WUFM031	WUFMSKM031	WUFCM031	WUFM131	WUFMSKM131	WUFCM131	INTTAUA0I2			

Caution Wake-up factors whose power domain is ISO0 can wake up Isolated area 0 or Isolated area 1 from STOP mode, but not from DEEPSTOP mode.

Table 10-4 Wake-up factor register assignments (WUFMm, WUFMSKMm, and WUFCMm; m = 0 or 1)

Assignment of WUFMm, WUFMSKMm, and WUFCMm register bits						Wake-up factors			
Isolated area 0 wake-up			Isolated area 1 wake-up			Wake-up factors	Module	Power domain	Clock domain
WUFH000	WUFMSKH000	WUFCH000	WUFH100	WUFMSKH100	WUFCH100	INTTAUA0I3	TAUA0	ISO0	ISO0_6
WUFH001	WUFMSKH001	WUFCH001	WUFH101	WUFMSKH101	WUFCH101	INTTAUA0I4			
WUFH002	WUFMSKH002	WUFCH002	WUFH102	WUFMSKH102	WUFCH102	INTTAUA0I5			
WUFH003	WUFMSKH003	WUFCH003	WUFH103	WUFMSKH103	WUFCH103	INTTAUA0I6			
WUFH004	WUFMSKH004	WUFCH004	WUFH104	WUFMSKH104	WUFCH104	INTTAUA0I7			
WUFH005	WUFMSKH005	WUFCH005	WUFH105	WUFMSKH105	WUFCH105	INTTAUA0I8			
WUFH006	WUFMSKH006	WUFCH006	WUFH106	WUFMSKH106	WUFCH106	INTTAUA0I9			
WUFH007	WUFMSKH007	WUFCH007	WUFH107	WUFMSKH107	WUFCH107	INTTAUA0I10			
WUFH008	WUFMSKH008	WUFCH008	WUFH108	WUFMSKH108	WUFCH108	INTTAUA0I11			
WUFH009	WUFMSKH009	WUFCH009	WUFH109	WUFMSKH109	WUFCH109	INTTAUA0I12			
WUFH010	WUFMSKH010	WUFCH010	WUFH110	WUFMSKH110	WUFCH110	INTTAUA0I13			
WUFH011	WUFMSKH011	WUFCH011	WUFH111	WUFMSKH111	WUFCH111	INTTAUA0I14			
WUFH012	WUFMSKH012	WUFCH012	WUFH112	WUFMSKH112	WUFCH112	INTTAUA0I15			
WUFH013	WUFMSKH013	WUFCH013	WUFH113	WUFMSKH113	WUFCH113	WDTA0TNMI	WDTA0	AWO	CKSCLK_A07
WUFH014	WUFMSKH014	WUFCH014	WUFH114	WUFMSKH114	WUFCH114	WDTA1TNMI	WDTA1	ISO0	CKSCLK_007

Caution Wake-up factors whose power domain is ISO0 can wake up Isolated area 0 or Isolated area 1 from STOP mode, but not from DEEPSTOP mode.

Wake-up by URTEnRX

Though the URTEnRx signals cannot be directly used as wake-up factors, external interrupts INTPx can be used instead, because URTEnRX and INTPx are alternative functions of the same pin.

10.2 Standby Controller Features

The microcontroller supports three power save modes:

- HALT mode
  HALT mode can be entered from normal operating mode by executing the CPU instruction HALT. This stops the CPU operation, while all clocks continue to operate and all areas remain powered.
- STOP mode
  In STOP mode, certain clock supplies of a clock domain can be stopped. STOP mode is entered by setting the clock stop bit STP of the relevant power save control register PSCm to 1. The clock selector of each clock domain can be configured to stop outputting a clock to its clock domain in STOP mode by clearing the CKSC_mn.STPMK_mn bit to 0. If this bit is set to 1, the output clock continues to be supplied to the clock domain, even if STOP mode is entered.
- DEEPSTOP mode
  To reduce power consumption further, the power supply of Isolated area 0 and Isolated area 1 can be switched off. To do so, set the power off bit PSCmPOF of the relevant power save control register PSCm (m = 0 or 1) to 1.

HALT HALT mode has no effect on power or clock domains. This mode is therefore not described in this chapter. For details about the HALT mode, see *V850E2M Architecture User's Manual*.

STOP/DEEPSTOP STOP mode and DEEPSTOP mode can be selected separately for different clock domains:

- the PSC0.PSC0STP bit controls the clock supply of the Always-On area and Isolated area 0
- the PSC1.PSC1STP bit controls the clock supply of Isolated area 1
- the PSC0.PSC0 bit controls the power supply of Isolated area 0
- the PSC1.PSC1POF bit controls the power supply of Isolated area 1

RUN All modes in which the CPU is operating are called RUN modes.

The clock and power supply options are summarized in the table below.

<R> **Table 10-5 Power-save modes overview**

Mode	Always-On	Isolated area 0		Isolated area 1	
	Clock	Power	Clock	Power	Clock
RUN	On	On	On	On	On
RUN (Iso1 STOP)	On	On	On	On PSC1.PSC1POF = 0	Stop PSC1.PSC1STP = 1
RUN (Iso1 DEEPSTOP)	On	On	On	Off PSC1.PSC1POF = 1	Stop PSC1.PSC1STP = 1
STOP	Stop PSC0.PSC0STP = 1	On PSC0.PSC0POF = 0	Stop PSC0.PSC0STP = 1	On PSC1.PSC1POF = 0	Stop PSC1.PSC1STP = 1
STOP (Iso1 power off)	Stop PSC0.PSC0STP = 1	On PSC0.PSC0POF = 0	Stop PSC0.PSC0STP = 1	Off PSC1.PSC1POF = 1	Stop PSC1.PSC1STP = 1
DEEPSTOP	Stop PSC0.PSC0STP = 1	Off PSC0.PSC0POF = 1	Stop PSC0.PSC0STP = 1	Off PSC1.PSC1POF = 1	Stop PSC1.PSC1STP = 1

Note that a certain clock domain is only stopped if clock stop is unmasked by clearing the CKSC_mn.STPMK_mn bit to 0

Caution Setting Isolated area 0 to DEEPSTOP mode (by setting the PSC0.PSC0POF bit to 1) while keeping Isolated area 1 operating is not permitted.

The following table shows the status of the various clock sources in the different modes.

<R> **Table 10-6 Clock sources in power-save modes**

Mode	Always-On				
	240 kHz IntOsc	8 MHz IntOsc	MainOsc	SubOsc	PLLk
RUN	Always running	Always running	Enabled	Enabled	Enabled
RUN (Iso1 STOP)		Always running	Enabled	Enabled	Enabled
RUN (Iso1 DEEPSTOP)		Always running	Enabled	Enabled	Enabled
STOP		Stopped	Stopped	Stopped	Stopped
STOP (Iso1 power off)		Stopped	Stopped	Stopped	Stopped
DEEPSTOP		Stopped	Stopped	Stopped	Stopped

Note that a certain clock source is only stopped if clock stop is unmasked by the corresponding stop request enable bit.

- MOSCE.MOSCSTPMK = 1 for the MainOsc
- ROSCE.ROSCSTPMK = 1 for the 8 MHz IntOsc
- PLLEk.PLLkSTPMK = 1 for PLLk

10.2.1 Wake-up

(1) Wake-up sources

The following wake-up events cause the system to exit a power save mode:

Table 10-7 Wake-up events

Mode	Reset	NMI	INTLVI	INTPx[a]	Peripheral modules operating on each power domain[ab]	CAN reception FCNnRX[a]	Soft ware	OCD
RUN (Iso1 STOP)	Yes	Yes	Yes	Yes	• All on AWO • All on Iso0	Yes	Yes	Yes
RUN (Iso1 DEEPSTOP)	Yes	Yes	Yes	Yes[c]	• All on AWO • All on Iso0	Yes	Yes	Yes
STOP mode	Yes	Yes	Yes	Yes	• Operating on AWO • Operating on Iso0	Yes	No	Yes
STOP (Iso1 DEEPSTOP)	Yes	Yes	Yes	Yes[c]	• Operating on AWO • Operating on Iso0	Yes	No	Yes
DEEPSTOP	Yes	Yes	Yes	Yes[c]	• Operating on AWO	Yes	No	Yes

a) The wake-up event must be enabled as a wake-up factor. See "*Wake-up factors*" in the wake-up factor tables in *10.1 "V850E2/Sx4-H STBC Features"*.

b) The peripheral module to be used as a wake-up event must be supplied with its clock from the clock controller. If its clock domain might be set to STOP mode, the standby mode request must be masked (by setting the CKSC_mn.STPMK_mn to 1), thus the clock for the peripheral module remains operating in standby mode.

c) To use the external interrupt INTPx as a wake-up event from DEEPSTOP mode, the INTPx analog filter must be set to edge detection mode (by clearing the FCLAnCTLm.FCLAnINTLm bit to 0). For details, see *2.6 "Description of Port Filters".*

HALT mode wake-up For details about the HALT mode, see *V850E2M Architecture User's Manual*.

External INTPx interrupts All external INTPx interrupts can terminate all power save modes. For details, see *Table 10-1 "Power save mode transitions"*.

CAN CRXDn The falling edge of the CAN reception signal FCNnRX can generate a wake-up from a power save mode. For details about CAN wake-up, see *22.10 "Power Saving Modes"*. If FCNnRX wakes the system up from DEEPSTOP mode, the FCNnRX interrupt service routine is not executed. In this case, FCNnRX is only used as a wake-up event and is not part of any data communication via the CAN interface.

Peripheral module interrupt Interrupts from a peripheral module can generate a wake-up, provided

the peripheral module is not located in a power domain that is in DEEPSTOP mode (power of this domain is switched off) and the peripheral module is supplied with its operating clock (for example, the module is located in a domain in STOP mode, but clock stop is masked by setting the CKSC_mn.STPMK_mn bit to 1, the peripheral module event is declared as a wake-up factor (see *10.2.1 "Wake-up"*), and the interrupt service routine for the peripheral interrupt is to be performed after wake-up, the interrupt must be unmasked (by setting the EICn.EIMKn bit to 1).

<R> **Software wake-up** A software wake-up from RUN (Iso1 STOP or Iso1 DEEPSTOP) mode can be triggered by setting the PSC1.PSC1ISOWU bit to 1.

(2) Wake-up control

Wake-up sources are selected by means of two sets of wake-up factor enable registers. These registers are used exclusively to control the wake-up of Isolated area 0 and Isolated area 1 from power save mode.

- Wake-up mask register: WUFMSKb0/1 (b = L, M, or H)
  Each bit of these registers is assigned to a wake-up source of Isolated area m. Wake-up by this source is enabled if its WUFMSKa.WUFMSKan bit is cleared to 0 (x = 0 to 31). When an enabled wake-up occurs, Isolated area 0/1 is woken up.
- Wake-up factor register: WUFb0/1 (b = L, M, or H)
  If Isolated area 0/1 has been woken up by a certain wake-up source, the corresponding wake-up source flag WUFm0/1.WUFmx is set to 1 (x = 0 to 31).
- Wake-up factor clear registers: WUFCb0/1 (b = L, M, or H)
  To reset the WUFa.WUFan flag in a wake-up factor register, its assigned bit WUFCa.WUFCan must be set to 1.

Note The wake-up source flags in the wake-up factor registers WUFm0/1 indicate only the occurrence of a valid wake-up event.
Thus an asserted wake-up source flag does not mean that a certain power domain has really changed from standby to operating mode.

(3) Main oscillator wake-up

MainOsc starts when a wake-up event occurs.
The control bits in the OSCWUFMSK register allow you to define which wake-up events can start the main oscillator after wake-up:

- OSCWUFMSK.OSCWUFMSK00 bit = 0
  enables the main oscillator to start operating after wake-up from Isolated area 0.
- OSCWUFMSK.OSCWUFMSK01 bit = 0
  enables the main oscillator to start operating after wake-up from Isolated area 1.

If either or both of these control bits are set to 1, the wake-up factors from the respective isolated area do not start the main oscillator.

Note The operation to automatically start up an oscillator after wake-up differs from the operation to automatically stop or start an oscillator by setting the MOSCE.MOSCESTPMK bit to 1.

- MOSCE.MOSCSTPMK bit = 1
  does not start an oscillator after wake-up if it was stopped before standby mode was entered.
- OSCWUFMSK0m bit = 0
  always starts an oscillator upon a wake-up event from the selected isolated area.

For details about the MOSCE.MOSCSTPMK bit, see *9.3.1 “Main oscillator (MainOsc) clock generator”* and *9.3.2 “Sub oscillator (SubOsc) clock generator”*.

<R> 10.2.2 I/O buffer control

This section describes the operation of the I/O buffers in various standby modes.

Caution All the I/O buffers of the V850E2/Sx4-H support the I/O buffer hold state. For details about the I/O buffers, see *Chapter 2 "Pin Functions"*.

(1) I/O buffer hold state

If an I/O buffer is set to the I/O buffer hold state, the state of the buffer is frozen. Thus its input or output buffer remains in the state before entering the I/O buffer hold state. No external or internal signal can change its state, until the I/O buffer hold state is terminated.

Besides the I/O buffer hold state during standby, the application software can also set all buffers of Isolated area m to the I/O buffer hold state by using the power save control registers PSCm:

- PSCm.PSCmIOHLDSET bit = 1:
  Isolated area m buffers enter I/O buffer hold state.
- PSCm.PSCmIOHLDSET bit = 0:
  Isolated area m buffers exit I/O buffer hold state.

(2) I/O buffers in STOP mode

The I/O buffers of the areas in STOP mode (clock has been stopped) maintain their settings. Because the peripheral modules or ports connected to the I/O buffers also remain in the state they were in before entering STOP mode, they continue to control the I/O buffers.

(3) I/O buffers in DEEPSTOP mode

The I/O buffers of the isolated area in DEEPSTOP mode enter the I/O buffer hold state by default, and so the buffer status is not changed.

After wake-up, the user application has to re-configure the peripheral module or port that generates the signals connected to a certain I/O buffer. Afterwards the I/O buffer hold state must be terminated by setting the PSCm.PSCmIOHLDCLR bit to 1. After setting this I/O buffer hold clear trigger, the relevant I/O buffer operates as configured by the peripheral module or port.

The automatic change to the I/O buffer hold state when entering DEEPSTOP mode can be controlled by using the power save control registers PSCm:

- PSCm.PSCmIOHLDMSK bit = 0:
  Isolated area m buffers *enter* I/O buffer hold state.
- PSCm.PSCmIOHLDMSK bit = 1:
  Isolated area m buffers *does not enter* I/O buffer hold state.

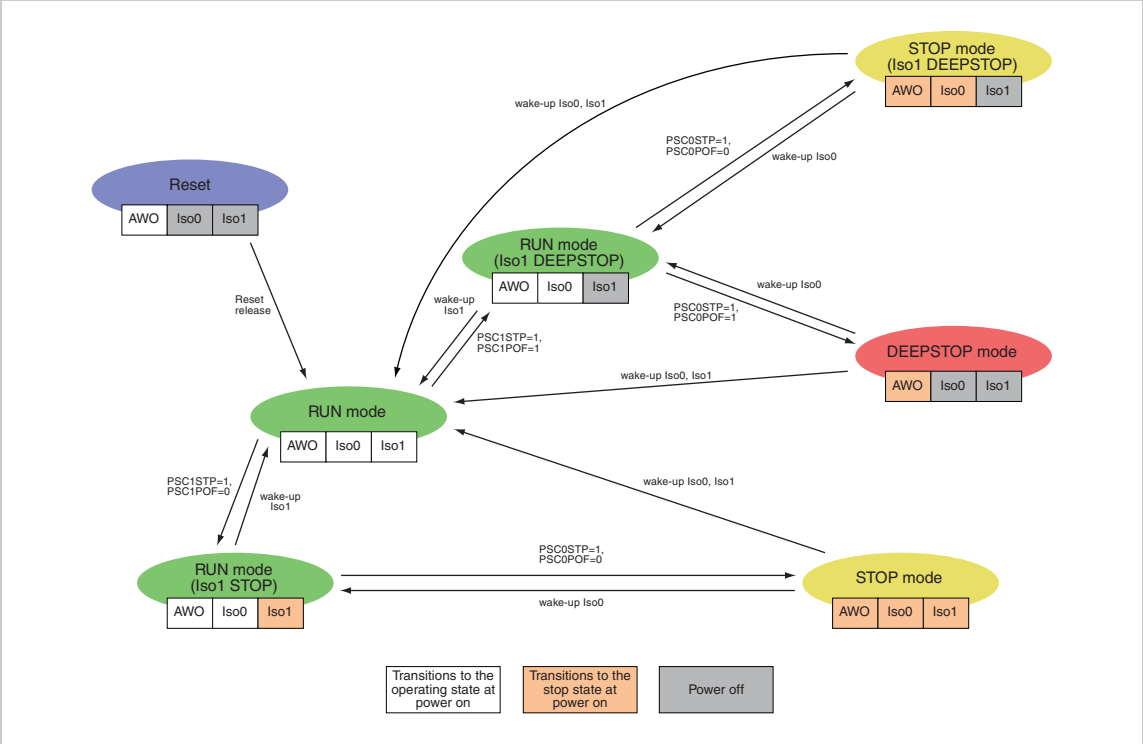
<R> **Table 10-8 Buffer operation in DEEPSTOP mode and after wake-up**

User action and states	Isolated area m power supply	
	On	Off (DEEPSTOP)
User action	I/O buffer power supply (no DEEPSTOP standby)	–
Buffer state during standby		I/O buffer hold state
Buffer state after wake-up from DEEPSTOP		I/O buffer hold state[a]

a) After wake-up and re-initialization of the Isolated area m modules, the I/O buffer hold state must be exited by setting the PSCm.PSCmIOHOLDCLR bit to 1.

10.2.3 Power save mode transitions

The following diagram shows the possible transitions between the various RUN and power save modes.



<R> **Figure 10-1 Power save mode transitions**

Caution Do not transition to a state in which the ISO0 power is off and the ISO1 power is on.

10.2.4 Examples of entering and exiting power save mode

The following examples show how to enter and exit power save modes.

Cautions

1. Before starting STOP mode preparations, make sure that all modules that are subject to the mode are not operating.

<R>

2. When stopping a clock source (main clock oscillator, subclock oscillator, or PLLk) by using the stop trigger bit (MOSCE.MOSCDISTRG, SOSCE.SOSCDISTRG, or PLLkE.PLLkDISTRG) before entering standby mode, either switch all of the clock sources used in the clock domain to other clock sources, or clear the CKSC_mn register to 0 to disable outputting the relevant clock.

(1) STOP mode

To enter STOP mode, stop the selected clocks of the modules in the Always-On area, Isolated area 0, and Isolated area 1.

The recommended procedure is as follows:

Standby preparation

- Stop all DMA channels.
- Disable interrupt servicing by executing the CPU instruction DI.
- Wake-up register settings
  - Clear the interrupt flag EICn.EIRFn to 0.
  - Mask non-wake-up interrupts (by setting the EICn.EIMKn bit to 1).
  - Unmask wake-up interrupts (by clearing the EICn.EIMKn bit to 0).
- WUFLm/WUFMm/WUFHm register settings
  - Clear wake-up factor flags (by setting the WUFCan bit of the WUFCLm, WUFCMm, or WUFCHm registers to 1).
  - Mask non-wake-up factors (by setting the WUFMSKan bit of the WUFMSKLm, WUFMSKMm, or WUFMSKHm register to 1).
  - Unmask non-wake-up factors (by clearing the WUFMSKan bit of the WUFMSKLm, WUFMSKMm, or WUFMSKHm register to 0).
- Set clock masks to select which clock domains will stop or continue operating. (To stop clock domain mn, clear the CKSC_mn.STPMK_mn bit to 0.)
- Select the clock generators to operate or stop during standby (by using the MOSCE.MOSCESTPMK, SOSCE.SOSCESTPMK, ROSCE.ROSCESTPMK, and PLLEk.PLLkSTPMK bits).

Entering standby

- Set PSC1 to 01_H to trigger transition to STOP mode for Isolated area 1.
- Set PSC0 to 01_H to trigger transition to STOP mode for the Always-On area and Isolated area 0.
- The microcontroller will enter STOP mode.

Exiting standby After a wake-up event occurs, make sure the PWS1.PWS1PSS bit is 0 by using the next instruction. If this bit is 0, it indicates that the system has exited STOP mode. The microcontroller resumes operating in this loop and exits the loop when wake-up is completed (indicated by the PWS1.PWS1PSS bit = 0).

Wake-up processing After enabling interrupt servicing by executing the CPU instruction EI, the wake-up interrupt is serviced.

<R>

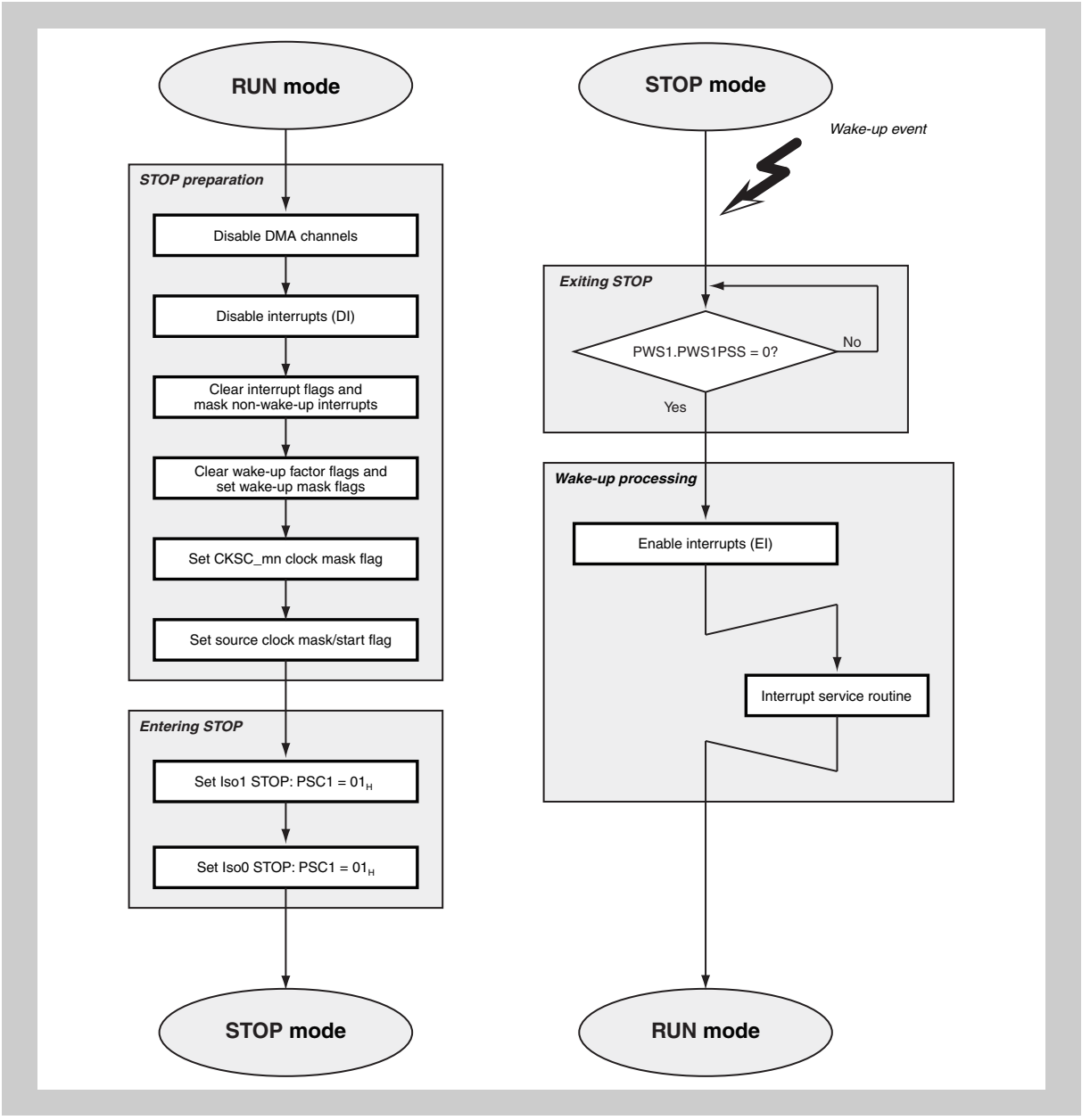


Figure 10-2 Recommended flow for entering and exiting STOP mode

(2) RUN mode (Iso1 STOP)

RUN mode (Iso1 STOP) is entered by stopping the selected clocks of the modules in Isolated area 1.

The recommended procedure is as follows:

Standby preparation

See "*STOP preparation*" in *(1) "STOP mode"*.

Entering standby

- Set PSC1 to 01_H to trigger transition to STOP mode for Isolated area 1.
- Check the wake-factor flags WUFa[31:0] of the WUFLm, WUFMm, and WUFHm registers. If any of the wake-up factor flags WUFa[31:0] is 1, a wake-up has occurred and the wake-up interrupt service routine can be performed after interrupts are enabled by executing the EI instruction.
- The PWS1.PWS1PSS bit is evaluated as follows:
  - If the PWS1.PWS1PSS bit is 0, the microcontroller has not completely entered RUN mode (Iso1 STOP), so check the wake-up factor flags.
  - If the PWS1.PWS1PSS bit is 1, Isolated area 1 has entered STOP mode.
- Enable interrupt servicing by executing the EI instruction.
- The microcontroller has entered RUN mode (Iso1 STOP) and the CPU is operating.

Exiting standby

After a wake-up event occurs, make sure the PWS1.PWS1PSS bit is 0 by using the next instruction. If this bit is 0, it indicates that the system has exited RUN mode (Iso1 STOP).
The microcontroller resumes operating in this loop and exits the loop when wake-up is completed (indicated by the PWS1.PWS1PSS bit is 0).

Wake-up processing

After enabling interrupt servicing by executing the CPU instruction EI, a wake-up interrupt is serviced.

Software wake-up

The RUN mode (Iso1 STOP) can also be terminated by setting the software wake-up trigger PSC1.PSC1ISOWU to 1.
In this case, no interrupt will be serviced.

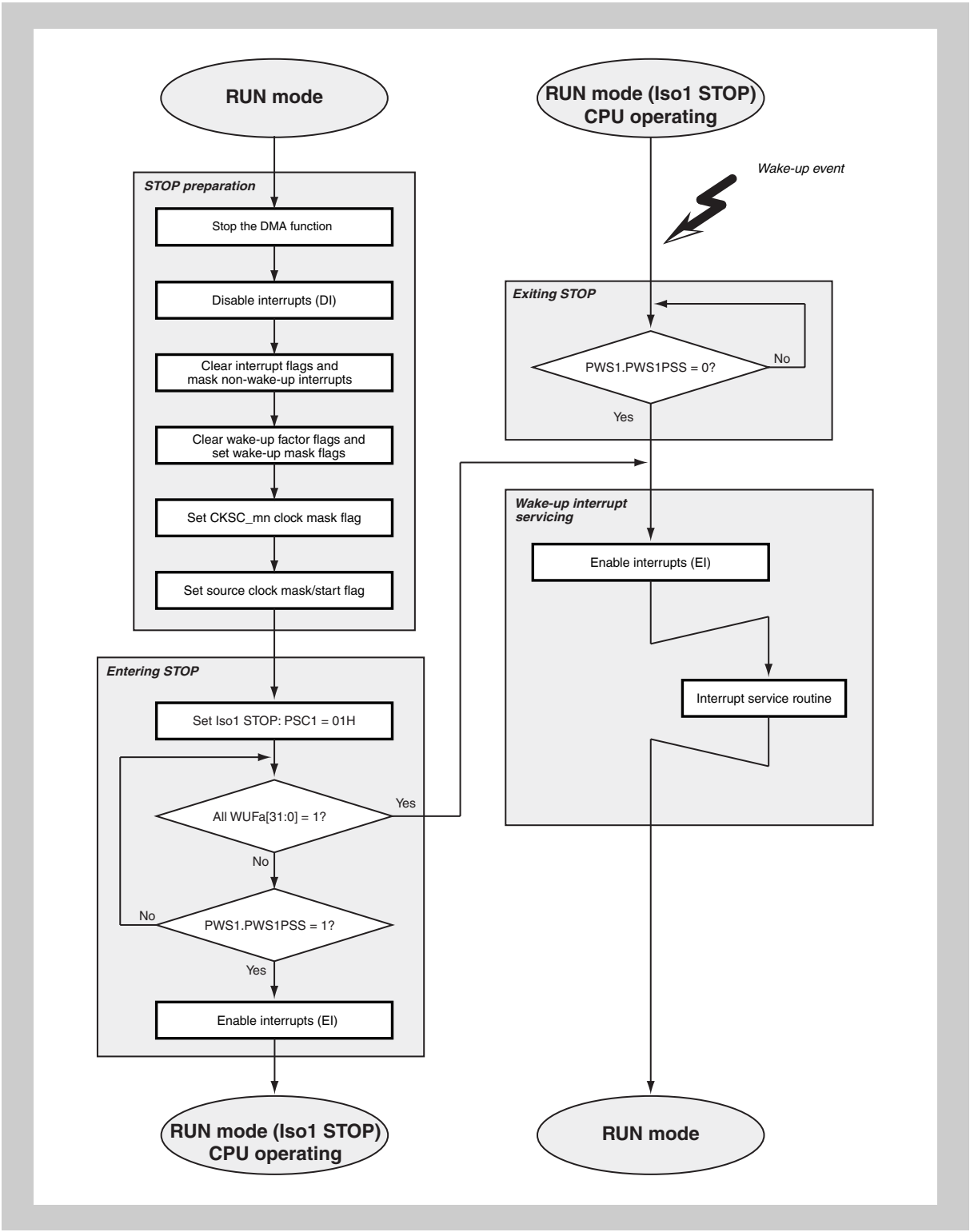


Figure 10-3 Recommended flow for entering and exiting RUN mode (Iso1 STOP)

(3) DEEPSTOP mode

In DEEPSTOP mode, the clock supply of all domains and the power supply of the isolated areas are stopped.

The recommended procedure is as follows:

Standby preparation

See "*STOP preparation*" in *(1) "STOP mode"*.

Entering standby

- Set PSC1 to 03_H to trigger transition to DEEPSTOP mode for Isolated area 1.
- Check the wake-up factor flags WUFa[31:0] of the WUFLm, WUFMm, and WUFHm registers.If any of the wake-up factor flags WUFa[31:0] is 1, a wake-up has occurred and wake-up processing can be performed.
- The PWS1.PWS1PSS and PWS1.PWS1ISO bits are evaluated as follows:
  - If the PWS1.PWS1PSS is not equal to 1 or the PWS1.PWS1ISO bit is not equal to 1, the microcontroller has not completely entered Iso1 DEEPSTOP mode.
  - If the PWS1.PWS1PSS and PWS1.PWS1ISO bits are equal to 1, the microcontroller has completely entered Iso1 DEEPSTOP mode.
- Set PSC0 to 03_H to trigger transition to DEEPSTOP mode for Isolated area 0.
- The microcontroller has entered DEEPSTOP mode.

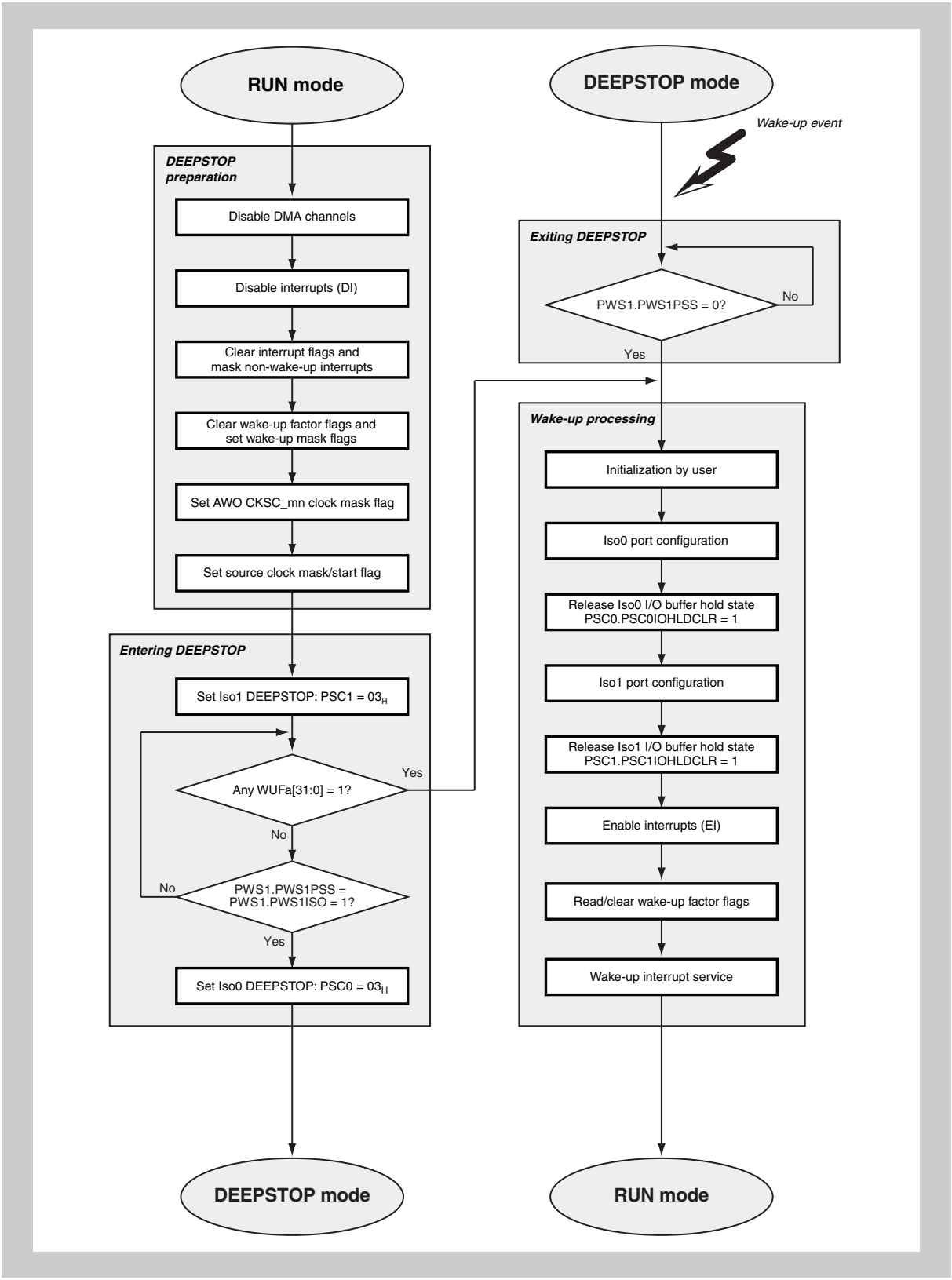
Exiting standby

After a wake-up event occurs, the microcontroller starts its reset procedure. Make sure the PWS1.PWS1PSS bit is 0 by using the next instruction. If this bit is 0, it indicates that the system has exited DEEPSTOP mode.
The microcontroller resumes operating in this loop and exits the loop when wake-up is completed (indicated by the PWS1.PWS1PSS bit is 0 and PWS1.PWS1ISO is 1).

<R> **Wake-up processing**

- Wake-up processing starts with user initialization (in the same way as after reset).
- The wake-up event is checked by using the wake-up factor flags, which can be cleared afterwards.
- Isolated area 0 port buffer operation is resumed by:
  - Setting the Iso0 ports
  - Releasing the Iso0 I/O buffer hold state (the PSC0.PSC0IOHLDCLR bit = 1)
- Isolated area 1 port buffer operation is resumed by:
  - Setting the Iso1 ports
  - Releasing the Iso1 I/O buffer hold state (the PSC1.PSC1IOHLDCLR bit = 1)

When Isolated area 0 and Isolated area 1 are in DEEPSTOP mode, the clock supply for all domains and the power supply for the isolated areas are stopped (by setting PSC0 = PSC1 = 03_H).



<R> **Figure 10-4 Recommended flow for entering and exiting DEEPSTOP mode**

(4) RUN mode (Iso1 DEEPSTOP)

To enter RUN mode (Iso1 DEEPSTOP), stop all the clock and power supplies of Isolated area 1.

The recommended procedure is as follows:

Standby preparation

- Stop all DMA channels.
- Disable interrupt servicing by executing the CPU instruction DI.
- Clear the interrupt flags EICn.EIRFn to 0.
  - Mask non-wake-up interrupts (by setting the EICn.EIMKn bit to 1).
  - Unmask wake-up interrupts (by clearing the EICn.EIMKn bit to 0).
- Clear wake-up factor flag registers (by setting WUFCan bit of the WUFCLm, WUFCMm, and WUFCHm registers to 1).
  - Mask non-wake-up factors (by setting the WUFMSKan bit of the WUFMSKLm, WUFMSKMm, and WUFMSKHm registers to 1).
  - Unmask non-wake-up factors (by clearing the WUFMSKan bit of the WUFMSKLm, WUFMSKMm, and WUFMSKHm registers to 0).

Entering standby

- Set PSC1 to 03_H to trigger transition to DEEPSTOP mode for Isolated area 1.
- Check the wake-up factor flags WUFa[31:0] of the WUFLm, WUFMm, and WUFHm registers.If any of the wake-up factor flags WUFa[31:0] is 1, a wake-up has occurred and the wake-up interrupt service routine can be performed after interrupts are enabled by executing the EI instruction.
- Check the PWS1.PWS1PSS and PWS1.PWS1ISO bits:
  - If the PWS1.PWS1PSS or PWS1.PWS1ISO bit is not equal to 1, the microcontroller has not completely entered Iso1 DEEPSTOP mode.
  - If the PWS1.PWS1PSS and PWS1.PWS1ISO bits are equal to 1, the microcontroller has completely entered Iso1 DEEPSTOP mode.
- Enable interrupt servicing by executing the EI instruction.
- The microcontroller has entered RUN mode (Iso1 DEEPSTOP).

Exiting standby

After a wake-up event occurs, the microcontroller starts its reset procedure. Make sure the PWS1.PWS1PSS and PWS1.PWS1ISO bits are 0 by using the next instruction. If this bit is 0, it indicates that the system has exited RUN mode (Iso1 DEEPSTOP). The microcontroller resumes operating in this loop and exits the loop when wake-up is completed (indicated by the PWS1.PWS1PSS and PWS1.PWS1ISO bits are 0).

Wake-up processing

- Wake-up processing starts with user initialization (in the same way as after reset).
- The wake-up event is checked by using the wake-up factor flags, which can be cleared afterwards.
- Isolated area 1 port buffer operation is resumed by:
  - Setting the Iso1 ports
  - Releasing the Iso1 buffer hold state (the PSC1.PSC1IOHLDCLR bit = 1)

After enabling interrupt servicing by executing the CPU instruction EI, a wake-up interrupt is serviced.

Software wake-up

The RUN mode (Iso1 DEEPSTOP) can be terminated by setting a software wake-up trigger PSC1.PSC1ISOWU to 1.
In this case, no interrupt will be serviced.

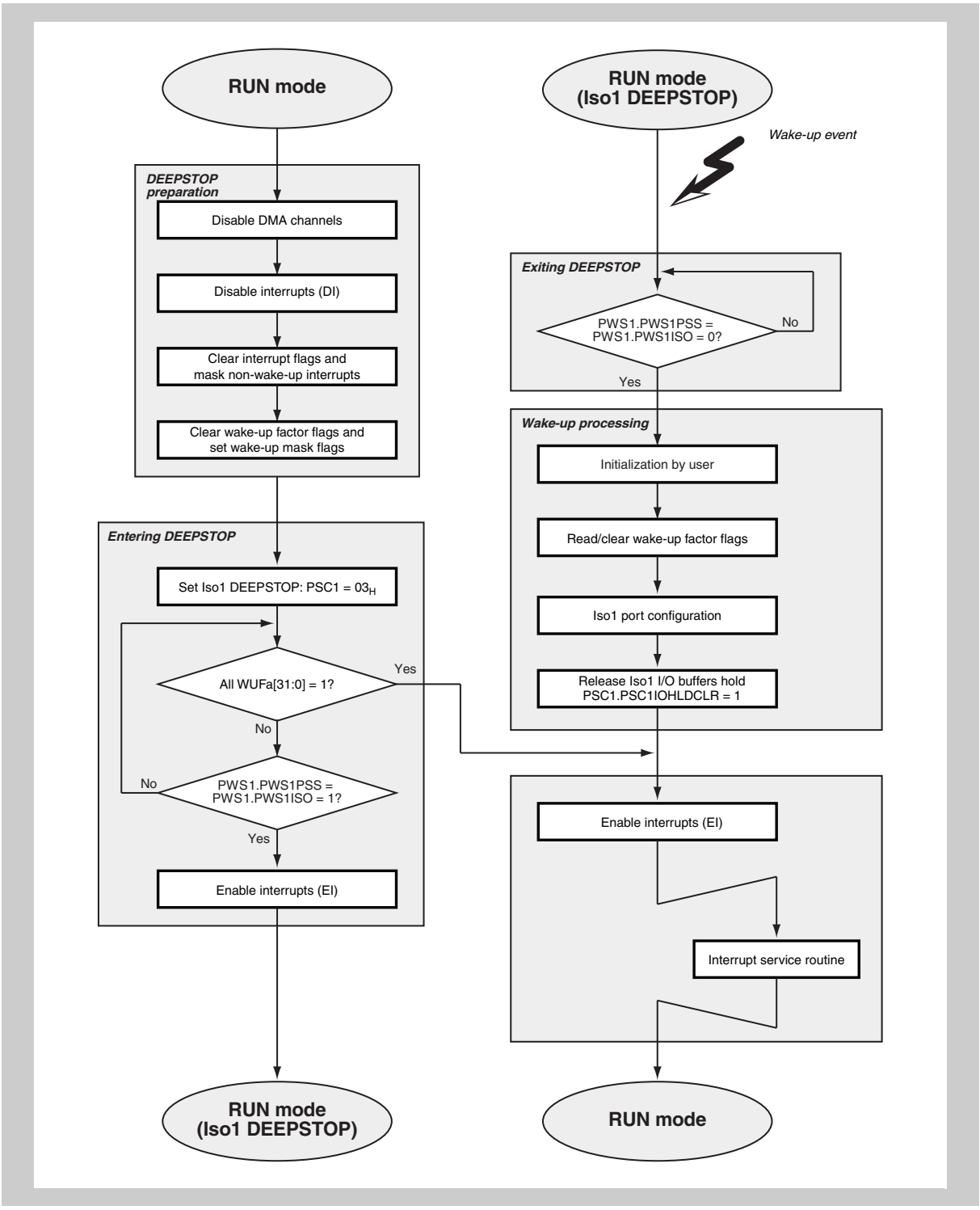


Figure 10-5 Recommended flow for entering and exiting RUN mode (Iso1 DEEPSTOP)

10.2.5 Writing to protected registers

Write protected registers are protected from inadvertent write access due to causes such as erroneous program execution.

The following standby controller registers feature this special write protection:

- Power save control registers PSC0 and PSC1.

For how to write to write-protected registers, see *3.10 “Write-Protected Registers”*.

10.3 Registers

10.3.1 Standby controller register overview

The STBC is controlled and operated by the following registers:

Table 10-9 Standby controller register overview

Register name	Symbol	Address
Power save registers		
Power save control register 0	PSC0	FF42 0000$_H$
Power save control register 1	PSC1	FF42 0008$_H$
Power status register 0	PWS0	FF42 0004$_H$
Power status register 1	PWS1	FF42 000C$_H$
Isolated area 0 wake-up event controller registers		
Wake-up factor register L	WUFL0	FF42 0100$_H$
Wake-up factor register M	WUFM0	FF42 0110$_H$
Wake-up factor register H	WUFH0	FF42 0120$_H$
Wake-up factor mask register L	WUFMSKL0	FF42 0104$_H$
Wake-up factor mask register M	WUFMSKM0	FF42 0114$_H$
Wake-up factor mask register H	WUFMSKH0	FF42 0124$_H$
Wake-up factor clear register L	WUFCL0	FF42 0108$_H$
Wake-up factor clear register M	WUFCM0	FF42 0118$_H$
Wake-up factor clear register H	WUFCH0	FF42 0128$_H$
Isolated area 1 wake-up event controller registers		
Wake-up factor register L	WUFL1	FF42 0130$_H$
Wake-up factor register M	WUFM1	FF42 0140$_H$
Wake-up factor register H	WUFH1	FF42 0150$_H$
Wake-up factor mask register L	WUFMSKL1	FF42 0134$_H$
Wake-up factor mask register M	WUFMSKM1	FF42 0144$_H$
Wake-up factor mask register H	WUFMSKH1	FF42 0154$_H$
Wake-up factor clear register L	WUFCL1	FF42 0138$_H$
Wake-up factor clear register M	WUFCM1	FF42 0148$_H$
Wake-up factor clear register H	WUFCH1	FF42 0158$_H$
Oscillator wake-up mask register:		
Oscillator wake-up mask register	OSCWUFMSK	FF42 01A4$_H$

10.3.2 Standby controller control register details

(1) PSC0 – Power save control register 0

This register is used to control the standby modes of the Always-On area and Isolated area 0.

Protection This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register PROTCMD2. For how to write to write-protected registers, see *3.10 "Write-Protected Registers"*.

Access This register can be read or written in 32-bit units.

Address FF42 0000$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	PSC0 IOHLDSET	PSC0 IOHLDMSK	PSC0 REGSTP	PSC0 IOHLDCLR	0	PSC0 POF	PSC0 STP
R	W	R/W	R/W	R/W	R	R/W	W

Table 10-10 PSC0 register contents (1/2)

Bit position	Bit name	Function
6	PSC0 IOHLDSET	Isolated area 0 I/O buffer hold state trigger 0: Terminate the Isolated area 0 I/O buffer hold state. 1: Enter the Isolated area 0 I/O buffer hold state. If the PSC0IOHLDSET bit is set to 1, the I/O buffers of Isolated area 0 enter the I/O buffer hold state. If the PSC0IOHLDSET bit is cleared to 0, the I/O buffers exit from I/O buffer hold state. The value of this bit is always 0 when read.
5	PSC0 IOHLDMSK	Isolated area 0 I/O buffer hold state mask 0: Enable the Isolated area 0 I/O buffer hold state. 1: Disable the Isolated area 0 I/O buffer hold state. If the I/O buffer hold state is disabled (PSC0IOHLDMSK = 1), the I/O buffers of Isolated area 0 do not enter the hold state in DEEPSTOP mode. The value of this bit is always 0 when read.
4	PSC0 REGSTP	Control of voltage regulators in DEEPSTOP mode, in combination with PSC1.PSC1REGSTP. See *Table 10-11 "Voltage regulator control" on page 524*.

Table 10-10 PSC0 register contents (2/2)

Bit position	Bit name	Function
3	PSC0 IOHLDCLR	Isolated area 0 I/O buffer hold state clear trigger 0: – 1: Clear the I/O buffer hold state. The PSC0IOHLDCLR bit can only be written with 1. Writing 0 is ignored. The value of this bit is always 0 when read.
1	PSC0 POF	Isolated area 0 power off selection for DEEPSTOP mode 0: Isolated area 0 power remains on when entering STOP mode. 1: Isolated area 0 power is switched off when entering STOP mode (DEEPSTOP). If the PSC0POF bit is set to 1, the system reset signal (SYSRES) is input to Isolated area 0 when wake-up occurs subsequently.
0	PSC0 STP	Always-On area and Isolated area 0 STOP mode trigger 0: – 1: Always-On area and Isolated area 0 enter STOP mode. The PSC0STP bit can only be written with 1. Writing 0 is ignored. The value of this bit is always 0 when read.

Caution Set the PSC1.PSC1POF bit to 1 before setting the PSC0.PSC0POF bit to 1.

Voltage regulator control

The PSC0.PSC0REGSTP and PSC1.PSC1REGSTP bits determine the operation of the internal voltage regulators and the WAKE output signal in DEEPSTOP mode:

Table 10-11 Voltage regulator control

Register settings		Voltage regulator control		
PSC0 REGSTP	PSC1 REGSTP	AWO power supply internal regulator	Iso0/Iso1 power supply internal regulator	WAKE output external regulator control
0	0	Normal operation	Normal operation	WAKE = 1 (external regulator must operate)
0	1			
1	0			
1	1	Low-voltage operation	Operation stopped	WAKE = 0 (external regulator can be switched off)

(2) PSC1 – Power save control register 1

This register is used to control the standby modes of Isolated area 1.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register PROTCMD2. For how to write to write-protected registers, see *3.10 "Write-Protected Registers"*.

Access This register can be read or written in 32-bit units.

Address FF42 0008$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	PSC1 IOHLDSET	PSC1 IOHLDMSK	PSC1 REGSTP	PSC1IOHL DCLR	PSC1I ISOWU	PSC1 POF	PSC1 STP
R	W	R/W	R/W	R/W	R/W	R/W	W

Table 10-12 PSC1 register contents (1/2)

Bit position	Bit name	Function
6	PSC1 IOHLDSET	Isolated area 1 I/O buffer hold state trigger 0: Terminate the Isolated area 1 I/O buffer hold state. 1: Enter the Isolated area 1 I/O buffer hold state. If the PSC1IOHLDSET bit is set to 1, the I/O buffers of Isolated area 1 enter the I/O buffer hold state. If the PSC1IOHLDSET bit is cleared to 0, the I/O buffers exit from I/O buffer hold state. Writing to this bit is ignored if Isolated area 1 is in standby mode (PWS1.PWS1PSS bit = 1). The value of this bit is always 0 when read.
5	PSC1 IOHLDMSK	Isolated area 1 I/O buffer hold state mask 0: Enable the Isolated area 1 I/O buffer hold state. 1: Disable the Isolated area 1 I/O buffer hold state. If the I/O buffer hold state is disabled (PSC1IOHLDMSK = 1), the I/O buffers of Isolated area 1 do not enter the hold state in DEEPSTOP mode. The value of this bit is always 0 when read.
4	PSC1 REGSTP	Control of voltage regulators in DEEPSTOP mode, in combination with the PSC0.PSC0REGSTP bit. See *Table 10-11 "Voltage regulator control" on page 524*.

Table 10-12 PSC1 register contents (2/2)

Bit position	Bit name	Function
3	PSC1 IOHLDCLR	I/O buffer hold clear trigger 0: – 1: Clear the I/O buffer hold state. Writing to this bit is ignored if Isolated area 1 is in standby mode (PWS1.PWS1PSS bit = 1). Writing 0 is ignored. The value of this bit is always 0 when read.
2	PSC1 ISOWU[a]	Isolated area 1 wake-up start trigger 0: – 1: Wake up Isolated area 1. The PSC1ISOWU bit can only be written with 1. Writing 0 is ignored. The value of this bit is always 0 when read.
1	PSC1 POF	Isolated area 1 power off selection for DEEPSTOP mode 0: Isolated area 1 power remains on when entering STOP mode 1: Isolated area 1 power is switched off when entering STOP mode (DEEPSTOP). If the PSC1POF bit is set to 1, the system reset signal (SYSRES) is input to Isolated area 1 when a wake-up occurs subsequently.
0	PSC1 STP	Isolated area 1 STOP mode trigger 0: – 1: Isolated area 1 enters STOP mode. The PSC1STP bit can only be written with 1. Writing 0 is ignored. The value of this bit is always 0 when read.

a) Unlike wake-up by other wake-up events whose occurrence can be recognized by checking the WUFa[31:0] flags of the WUFLm, WUFMm, and WUFHm registers, a software wake-up by setting the PSC1ISOWU bit to 1 cannot be recognized by any flag.

(3) PWS0 – Power status register 0

This register shows the status of the standby mode of Isolated area 0.

Access This register is read-only, in 32-bit units.

Address FF42 0004$_H$

Initial value 0000 0001$_H$

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	0	0	0	0	0	PWS0 IOHOLD	1
R	R	R	R	R	R	R	R

Table 10-13 PWS0 register contents

Bit position	Bit name	Function
1	PWS0 IOHOLD	Isolated area 0 I/O buffer hold status 0: I/O buffers of Isolated area 0 is not in the hold state. 1: I/O buffers of Isolated area 0 is in the hold state.

Note This register cannot be read when Isolated area 0 is in STOP or DEEPSTOP mode.

(4) PWS1 – Power status register 1

This register shows status of the standby mode of Isolated area 1.

Access This register is read-only, in 32-bit units.

Address FF42 000C$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
PWS1 PSS	0	0	0	0	0	PWS1 IOHOLD	PWS1 ISO
R	R	R	R	R	R	R	R

Table 10-14 PWS1 register contents

Bit position	Bit name	Function
7	PWS1PSS	Isolated area 1 power status 0: Isolated area 1 is in normal operation mode. 1: Isolated area 1 is in one of the following statuses: - In the transition period from when the PSC1.POF bit is set to when the clock is stopped - The clock supply is stopped - In the transition period from when wake-up is received to when the clock starts operating
1	PWS1 IOHOLD	Isolated area 1 I/O buffer hold status 0: I/O buffers of Isolated area 1 is in the hold state. 1: I/O buffers of Isolated area 1 is in the hold state.
0	PWS1ISO	Isolated area 1 standby mode status 0: Isolated area 1 is in the power-off state. 1: Isolated area 1 is not in the power-off state.

10.3.3 Wake-up event controller register details

(1) WUFLm/WUFMm/WUFHm – Wake-up factor register (m = 0 or 1)

This register reports the wake-up events of Isolated area m.

Access This register is read-only, in 32-bit units.

Address WUFL0: FF42 0100_H
WUFM0: FF42 0110_H,
WUFH0: FF42 0120_H
WUFL1: FF42 0130_H,
WUFM1: FF42 0140_H
WUFH1: FF42 0150_H

Initial value $0000\ 0000_H$. This register is initialized by the power-up reset PURES.

31	30	29	28	27	26	25	24
WUFa31	WUFa30	WUFa29	WUFa28	WUFa27	WUFa26	WUFa25	WUFa24
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
WUFa23	WUFa22	WUFa21	WUFa20	WUFa19	WUFa18	WUFa17	WUFa16
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
WUFa15	WUFa14	WUFa13	WUFa12	WUFa11	WUFa10	WUFa9	WUFa8
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
WUFa7	WUFa6	WUFa5	WUFa4	WUFa3	WUFa2	WUFa1	WUFa0
R	R	R	R	R	R	R	R

Table 10-15 WUFLm/WUFMm/WUFHm register contents

Bit position	Bit name	Function
31 to 0	WUFan	Indicates the occurrence of the wake-up event WUan (a = L0, L1, M0, M1, H0, or H1, n = 0 to 31). 0: Wake-up event WUan did not occur. 1: Wake-up event WUan occurred.

Caution When the WUFMSKan bit is 1, the WUFan bit is not set even if a wake-up factor has occurred.

Wake-up factors For details about the assignment of wake-up events to wake-up factor register bits, see *10.2.1 "Wake-up"*.

When a bit that is assigned to a wake-up event is read, the value read is 0.

(2) WUFMSKLm/WUFMSKMm/WUFMSKHm – Wake-up factor mask registers (m = 0 or 1)

This register is used to enable the wake-up events of Isolated area m.

Access This register can be read or written in 32-bit units.

Address WUFMSKL0: FF42 0104$_H$
WUFMSKM0: FF42 0114$_H$,
WUFMSKH0: FF42 0124$_H$
WUFMSKL1: FF42 0134$_H$,
WUFMSKM1: FF42 0144$_H$
WUFMSKH1: FF42 0154$_H$

Initial value FFFF FFFF$_H$

31	30	29	28	27	26	25	24
WUF MSKa31	WUF MSKa30	WUF MSKa29	WUF MSKa28	WUF MSKa27	WUF MSKa26	WUF MSKa25	WUF MSKa24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
WUF MSKa23	WUF MSKa22	WUF MSKa21	WUF MSKa20	WUF MSKa19	WUF MSKa18	WUF MSKa17	WUF MSKa16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
WUF MSKa15	WUF MSKa14	WUF MSKa13	WUF MSKa12	WUF MSKa11	WUF MSKa10	WUF MSKa9	WUF MSKa8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
WUF MSKa7	WUF MSKa6	WUF MSKa5	WUF MSKa4	WUF MSKa3	WUF MSKa2	WUF MSKa1	WUF MSKa0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 10-16 WUFMSKLm/WUFMSKMm/WUFMSKHm register contents

Bit position	Bit name	Function
31 to 0	WUF MSKan	Enable the wake-up factor WUan (a = L0, L1, M0, M1, H0, or H1, n = 0 to 31) 0: Wake-up factor WUan enabled 1: Wake-up factor WUan disabled

Cautions

1. When the WUFMSKan bit is 1, the WUFan bit is not set even if a wake-up factor has occurred.
2. When transitioning to DEEPSTOP mode, the combination of setting the WUFMSKb0n bit to 1 and clearing the WUFMSKb1n bit to 0 is prohibited.

Wake-up factors For details about the assignment of wake-up events to wake-up factor register bits, see *"Wake-up factors" on page 500*.

When writing to this register, write 1 to a bit that has not been assigned to a wake-up event.

(3) WUFCLm/WUFCMm/WUFCHm – Wake-up factor clear registers (m = 0 or 1)

This register is used to clear the wake-up event indicated by the WUFbm register of Isolated area m (b = L, M, or H).

Access This register can be written in 32-bit units.
The value of this bit is always 0000 0000_H when read.

Address WUFCL0: FF42 0108_H
WUFCM0: FF42 0118_H,
WUFCH0: FF42 0128_H
WUFCL1: FF42 0138_H,
WUFCM1: FF42 0148_H
WUFCH1: FF42 0158_H

Initial value 0000 0000_H

31	30	29	28	27	26	25	24
WUFCa31	WUFCa30	WUFCa29	WUFCa28	WUFCa27	WUFCa26	WUFCa25	WUFCa24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
WUFCa23	WUFCa22	WUFCa21	WUFCa20	WUFCa19	WUFCa18	WUFCa17	WUFCa16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
WUFCa15	WUFCa14	WUFCa13	WUFCa12	WUFCa11	WUFCa10	WUFCa9	WUFCa8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
WUFCa7	WUFCa6	WUFCa5	WUFCa4	WUFCa3	WUFCa2	WUFCa1	WUFCa0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 10-17 WUFCLm/WUFCMm/WUFCHm register contents

Bit position	Bit name	Function
31 to 0	WUFCan	Clear the wake-up factor WUFan of the wake-up factor registers (a = L0, L1, M0, M1, H0, or H1, n = 0 to 31). 0: Do not change WUFan. 1: Clear WUFan.

Wake-up factors For details about the assignment of wake-up events to wake-up factor register bits, see *"Wake-up factors" on page 500*.

When writing to this register, write 0 to a bit that has not been assigned to a wake-up event.

10.3.4 Oscillator wake-up mask register details

(1) OSCWUFMSK – Oscillator wake-up mask register

This register is used to control the start of MainOsc when a wake-up event for Isolated area 0 or Isolated area 1 occurs.

Access This register can be read or written in 32-bit units.

Address FF42 01A4$_H$

Initial value 0000 0003$_H$. This register is initialized by the power-up reset PURES.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	0	0	0	0	0	OSCWUFMSK01	OSCWUFMSK00
R	R	R	R	R	R	R/W	R/W

Table 10-18 OSCWUFMSK register contents

Bit position	Bit name	Function
1	OSCWUFMSK01	Enable the Isolated area 1 wake-up factor for starting MainOsc. 0: The main oscillator is started by the Isolated area 1 wake-up factor. 1: The main oscillator is not started by the Isolated area 1 wake-up factor.
0	OSCWUFMSK00	Enable the Isolated area 0 wake-up factor for starting MainOsc. 0: The main oscillator is started by the Isolated area 0 wake-up factor. 1: The main oscillator is not started by the Isolated area 0 wake-up factor.

Note If the OSCWUFMSK0m bit is 0, MainOsc always starts when a wake-up from the isolated area occurs even if MainOsc was stopped before the standby mode was entered.

Chapter 11 Code Protection and Security

11.1 Overview

The microcontroller supports various methods for protecting the program code in the flash memory from undesired access, such as illegal read-out or illegal reprogramming.

In general, interfaces offer access to the internal flash memory: Nexus debug interface, external flash programmer interface, self-programming facilities and test interfaces.

In the following the security relevant items are listed. The features to protect the internal flash memory data from being read by unauthorized persons are described.

For details about the flash memory, see *Chapter 7 “Flash Memory”*.

The following sections give an overview about supported code protection methods.

11.2 Flash Programmer and Self-Programming Protection

In general, illegal read-out and re-programming of the flash memory is possible via the flash writer interface and the self-programming feature. For protection of the flash memory, the following flags provide various protection levels.

The flags can be set by flash programmers. For details about the flash memory programming, see *Chapter 7 "Flash Memory"*.

(1) Program protection flag (Program protection function)

Set this flag to disable the programming function via flash writer interface. This flag does not affect the self-programming interface.

The flag is valid for the whole flash memory.

(2) Chip erase protection flag (Chip erase protection function)

Set this flag to disable the chip erase function via flash memory programmer interface. This flag does not affect the self-programming interface.

(3) Block erase protection flag (Block erase protection function)

Set this flag to disable the feature to erase single blocks via flash writer interface. This flag does not affect the self-programming interface.

This flag does not affect the chip erase function.

The flag is valid for the whole flash memory.

(4) Read-out protection flag (Read-out protection function)

Set this flag to disable the feature that allows reading back the flash memory via flash writer interface. This flag does not affect the self-programming interface.

The flag is valid for the whole flash memory.

(5) Boot block cluster protection flag

Set this flag to disable erasure and rewrite of the boot block cluster.The boot block cluster can not be manipulated in any way (no erase/write).

This applies in serial and self-programming mode.

Once this flag is set, it is impossible to reset this flag. Thus the boot block cluster content can not be changed any more.

(6) Flash shielding

Flash shielding specifies a flash memory area, the programming window, that is permitted to be erased and programmed. Writing to and erasing of the remaining flash memory areas outside the programming window is impossible.

11.3 On-Chip Debug Interface Protection

In general, illegal read-out of the flash memory is possible via the Nexus on-chip debug interface. For protection of the flash memory, the usage of the debug interface can be disabled.

The debug interface is protected via a 95-bit ID code and an internal control flag (on-chip debugging enable control).

When the debugger is started, the status of the control flag is queried. Setting this flag to zero disables the on-chip debugger.

When debugging is enabled (on-chip debugging enable flag is set), you have to enter the 95-bit ID code via the debugger. The code is compared with the ID code stored in the internal flash memory. If the codes do not match, debugging is not possible.

The on-chip debugging enable flag and the ID code are stored in the flash memory extra area and can be accessed via the OCDSIDL, OCDIDM, OCDIDH registers.

11.3.1 On-chip debugging enable flag

On-chip debugging is controlled by the on-chip debugging enable flag OCDIDH.OCDID[95].

The on-chip debugging enable flag can be set or reset under the following circumstances.

- while reprogramming the flash memory by an external flash programmer
- while reprogramming the flash memory by an external flash programmer and self programming.
- by the user's program with the self-programming feature.

Enabling and disabling on-chip debugging is done by the on-chip debug control register IDMODI:

- Setting IDMODI.IDEN to 1 and clearing IDMODI.IDDATA to 0 clear OCDIDH.OCDID[95] to 0 to disable on-chip debugging.
- Setting IDMODI.IDEN to 1 and IDMODI.IDDATA to 1 set OCDIDH.OCDID[95] to 1 to enable on-chip debugging.

Notes

1. After release from the next external $\overline{\text{RESET}}$ or power-on-clear reset POCRES (for M1 products), the on-chip debug control bit OCDIDH.OCDID[95] takes on the value from the internal flash memory.
2. Writing to IDMODI with IDMODI.IDEN = 0 does not modify the on-chip debug control bit.

11.3.2 On-chip debug ID code

The 95-bit ID code is accessible via the register bits OCDIDH.OCDID[94:64], OCDIDM.OCDID[63:32] and OCDIDL.OCDID[31:0].

The ID code can be specified

- while reprogramming the flash memory by an external flash programmer
- while reprogramming the flash memory by an external flash programmer and self programming.

11.3.3 On-chip debug protection levels summary

The following table summarizes the protection levels of the on-chip debug interface:

Table 11-1 On-chip debug protection levels

<R>

On-chip debugging enable flag	ID code	Protection level
0	X[a]	Level 2: Full protection The on-chip debug interface cannot be used.
1	user-specific ID code	Level 1: ID code protection The on-chip debug interface can only be used if the user enters the correct ID code.
	ID code is all ones[b]	Level 0: No protection The on-chip debug interface can be used.

a) ID codes are not compared
b) This is the default state after the flash memory has been erased.

Notes
1. Once the on-chip debug interface has been set as “use-prohibited”, it cannot be used until the on-chip debugging enable flag is set to 0 by the user’s program or by self-programming.
2. After you have set protection levels 1 or 2, set the “block erase disable flag” in the flash extra area. Otherwise, an unauthorized person could erase the block that contains the ID code or the “on-chip debugging enable flag”, respectively, and thus suspend the protection.

11.3.4 On-chip debug control register

The following registers are dedicated to the on-chip debugger:

Table 11-2 Flash mask options register overview

Register name	Symbol	Address
On-chip debug ID register L	OCDIDL	FF47 0000_H
On-chip debug ID register M	OCDIDM	FF47 0004_H
On-chip debug ID register H	OCDIDH	FF47 0008_H
On-chip debug control register	IDMODI	FF47 0000_H

(1) OCDIDL/M/H - On-chip debug ID register

These registers hold the 95-bit ID code, the user is requested to enter upon start of a debug session.
By using the OCDID[95] bit, the on-chip debugging can generally be disabled or enabled.

Access In normal operation mode, this register is read-only, in 32-bit units.
Writing to this register is only possible in flash programming and self-programming mode.
The on-chip debug control bit OCDID[95] can be temporarily modified via the IDMODI register also in normal operation mode.

Address OCDIDL: FF47 0000_H, OCDIDM: FF47 0004_H, OCDIDH: FF47 0008_H

Initial value User-defined value

OCDIDH:

31	30	...		0
OCDID [95]	OCDID [94]	...	OCDID [65]	OCDID [64]
R	R	...	R	R

OCDIDM:

31	30	...		0
OCDID [63]	OCDID [62]	...	OCDID [33]	OCDID [32]
R	R	...	R	R

OCDIDL:

31	30	...		0
OCDID [31]	OCDID [30]	...	OCDID [1]	OCDID [0]
R	R	...	R	R

Table 11-3 OCDIDH/M/L register contents

Register	Bit position	Bit name	Function
OCDIDH	31	OCDID[95]	Enable/disable on-chip debug: 0: On-chip debug disabled 1: On-chip debug enabled
OCDIDH	30 to 0	OCDID[94:64]	95-bit on-chip debug ID code
OCDIDM	31 to 0	OCDID[63:32]	
OCDIDL	31 to 0	OCDID[31:0]	

(2) IDMODI - On-chip debug control register

This register allows to temporarily enable/disable on-chip debugging in normal operation mode, i.e. by the user's software.
This is performed by modifying the on-chip debug control bit OCDIDH.OCDID[95].

Note After release from the next external $\overline{\text{RESET}}$ or power-on-clear reset POCRES (for M1 products), the on-chip debug control bit OCDIDH.OCDID[95] takes on the value from the internal flash memory.

<R> **Protection** This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register PROTCMD3.
For how to write to write-protected registers, see *3.10 "Write-Protected Registers"*.

<R> **Access** This register can be read or written in 8-bit units.
Reading this register returns the value of OCDIDL.

Address FF47 0000$_H$

Initial value User-defined value

<R>

7	6	5	4	3	2	1	0
0	0	0	0	0	0	IDEN	IDDATA
W	W	W	W	W	W	R/W	R/W

Table 11-4 IDMODI register contents

Bit position	Bit name	Function
1, 0	IDEN, IDDATA	Enable or disable the on-chip debug control bit OCDIH.OCDI[95].

IDEN	IDDATA	Changing OCDIH.OCDI[95]
0	X	Not changed
1	0	Clear to 0 (on-chip debugging disabled)
	1	Set to 1 (on-chip debugging enabled)

11.4 Flash Programmer and Self-Programming Protection

In general, illegal read-out from and re-programming the flash memory contents are possible via the flash writer interface and the self-programming feature. To protect the flash memory, the flags below provide various protection levels.

The flags can be set by using flash programmers. For details about flash memory programming, see *Chapter 7 "Flash Memory"*.

(1) Chip erase protection flag (Chip erase protection function)

Set this flag to disable chip-erasing vis the flash memory programmer interface.

This flag does not affect the self-programming interface.

(2) Block erase protection flag (Block erase protection function)

Set this flag to disable erasing single blocks via the flash memory programmer interface.

This flag does not affect the self-programming interface.

This flag does not affect the chip erase function.

The flag is valid for the whole flash memory.

(3) Program protection flag (Program protection function)

Set this flag to disable programming via the flash memory programmer interface. This flag does not affect the self-programming interface.

The flag is valid for the whole flash memory.

(4) Read-out protection flag (Read-out protection function)

Set this flag to disable reading the flash memory via flash memory programmer interface.

This flag does not affect the self-programming interface.

The flag is valid for the whole flash memory.

(5) Boot block cluster protection flag

Set this flag to disable erasuring and rewriting the boot block cluster.
The boot block cluster cannot be manipulated in any way (no erase/write).

This flag can be used in serial and self-programming modes.

Once this flag is set, this flag cannot be reset. Thus the boot block cluster content can no longer be changed.

(6) Flash shielding

Flash shielding allows to specify a random number of consecutive code flash memory blocks, so the flash shield window, which can be erased and written

Flash shielding specifies a flash memory area, the programming window, that is permitted to be erased and programmed. Erasuring and writing to all flash memory blocks outside the flash shield window is impossible.

Chapter 12 Reset Controller

12.1 Functional Overview

Several system reset functions are provided in order to initialize the microcontroller hardware and its registers.

Feature summary

A reset can be triggered by the following events:

- External reset signal $\overline{\text{RESET}}$
  Noise in the external reset signal is eliminated by an analog filter.
- Power-on clear (POCRES)[a]
- Overflow of the watchdog timers (WDTA0RES, WDTA1RES)
- Clock monitor reset ($\overline{\text{CLMA0RES}}$, $\overline{\text{CLMA2RES}}$, $\overline{\text{CLMA3RES}}$)
- Low voltage indicator reset ($\overline{\text{LVIRES}}$)
- Software reset (SWRES)
- Debugger reset (DBRES)

a) For M1 products

The following block diagram shows the main components of the reset controller.

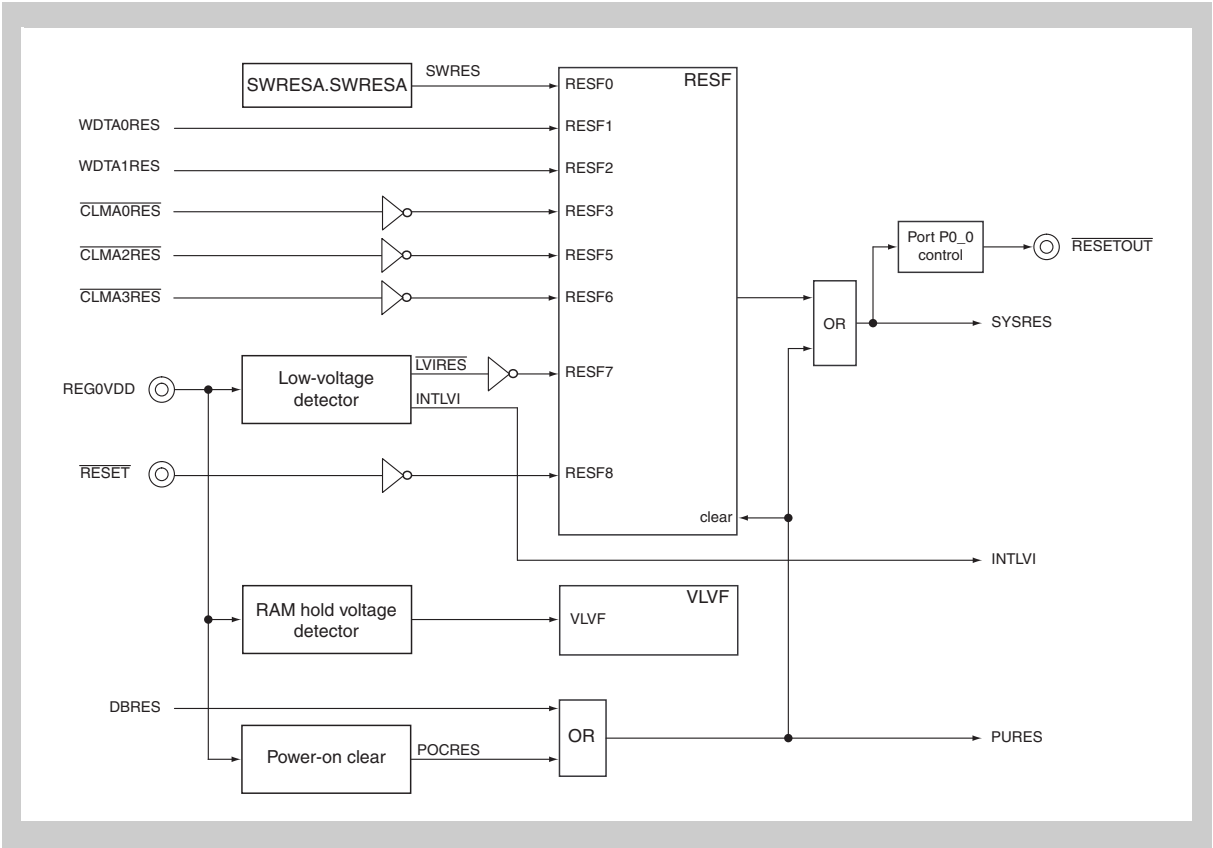


Figure 12-1 Block diagram of the reset controller: M1 products

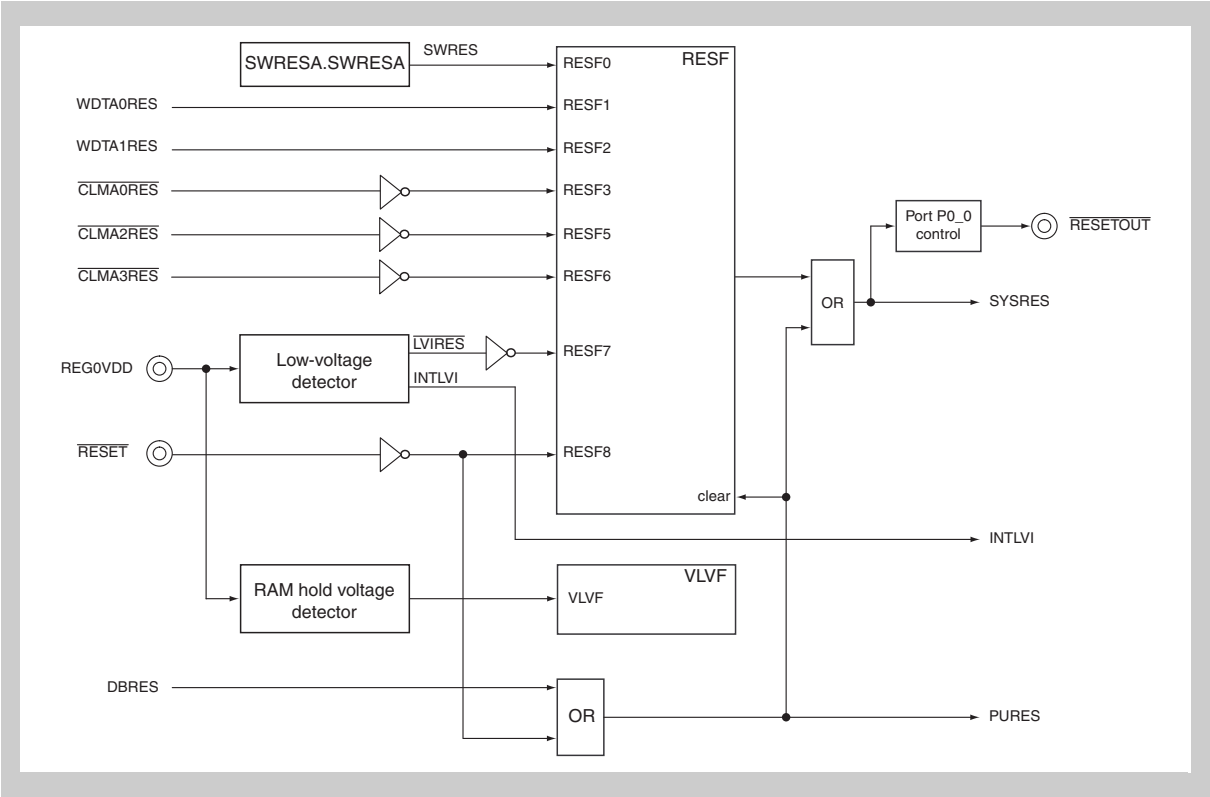


Figure 12-2 Block diagram of the reset controller: M2 products

(1) Reset signals

The reset controller manages the generation of two reset signals upon the occurrence of reset signals from various reset sources:

- System reset SYSRES
  The system reset is generated by all reset sources. SYSRES is applied to all microcontroller components, except the clock generators. Consequently, all clock generators continue operating, provided they were operating before SYSRES occurs.
- Power-up reset PURESThe power-up reset signal PURES is asserted by the power-on-clear reset POCRES (for M1 products), an external $\overline{\text{RESET}}$ signal (for M2 products), or the debugger reset signal DBRES. Because the clock generators are reset when PURES becomes active, the clock generators must be restarted after are stopped by PURES. For details about clock generators stopped by PURES, see *Chapter 9 "Clock Controller"*.
  In addition, PURES can be asserted by the system reset signal SYSRES. In this case, all microcontroller components including the clock generators are reset.

(2) Reset flag

The reset factor register RESF has flags for each reset source. If a reset source is asserted, the respective flag is set.The reset flags can only be cleared by input of the power-on-clear reset POCRES (for M1 products), $\overline{\text{RESET}}$ pin input (for M2 products), or by software.
For details, see *12.2.1 "Reset flag"*.

(3) On-chip module resets

Watchdog timer reset and WDTATRTYP

The watchdog timers can generate the reset signals WDTA0RES and WDTA1RES.Whether the watchdog timer continues or stops operating when the WDTATRTYP signal is input depends on the reset source.
For details, see *12.2.6 "Watchdog timer reset"*.

Clock monitor resets

The clock monitor can generate the resets signals $\overline{\text{CLMA0RES}}$, $\overline{\text{CLMA2RES}}$, and $\overline{\text{CLMA3RES}}$.
For details, see *12.2.8 "Clock monitor reset"*.

Debugger reset

A debugger reset is generated when a debugger is connected. When a debugger reset occurs, the power-up reset PURES also occurs.

(4) Software controlled resets

SWRES

A software reset can be generated by setting the software reset control register SWRESA.
For details, see *12.2.7 "Software reset"*.

(5) Reset output signal

After a reset ends, the P0_0 pin outputs an active $\overline{\text{RESETOUT}}$ signal.

Any change of the P0_0 configuration terminates the $\overline{\text{RESETOUT}}$ output.

(6) Power supply supervision

Several circuits observe the level of the external power supply REG0VDD and generate different actions according to its level.

Low voltage indicator

The low voltage indicator (LVI) generates the $\overline{\text{LVIRES}}$ reset if the voltage level of REG0VDD drops below a certain level. The level can be adjusted and $\overline{\text{LVIRES}}$ can be masked.
For details, see *12.2.3 “Low-voltage indicator (LVI)”*.

Power-on clear

The power-on-clear circuit (POC) constantly compares the power supply voltage REG0VDD with an internal reference voltage. This ensures that the microcontroller only operates as long as the power supply exceeds a certain limit.
For details, see *12.2.2 “Power-on clear (POC)”*.

RAM retention voltage indicator

The VLVF.VLVF flag for the RAM retention voltage detector RAMHF indicates that REG0VDD has dropped below the level at which retention of the backup RAM (BURAM) content is not guaranteed.
For details, see *12.2.4 "RAM retention voltage indicator (RAMHF)"*.

12.2 Functional Description

12.2.1 Reset flag

The reset factor register RESF provides reset flags for each reset source.

If a reset occurs, the corresponding flag is set. The reset source can be determined by checking this flag.

The RESF register flags can only be cleared by the power-on-clear reset POCRES (for M1 products), by $\overline{\text{RESET}}$ pin input (for M2 products), or by software. The value of each flag is retained in all other cases. Each reset source sets its corresponding flag, independent of other reset sources.

12.2.2 Power-on clear (POC)

Caution M1 products have a power-on-clear circuit (POC).
M2 products do not have a power-on-clear circuit (POC).

The power-on-clear circuit (POC) constantly compares the power supply voltage REG0VDD with an internal reference voltage V_{POC}. This ensures that the microcontroller only operates as long as the power supply exceeds a certain limit.

If REG0VDD < V_{POC} drops below the internal reference voltage, the internal reset signal POCRES, the system reset signal SYSRES, and the power-up reset signal PURES are generated.

For details about the specifications of the internal reference voltage level, see the Data Sheet.

At a power-on clear reset, the reset factor register RESF is cleared.

When the POCRES signal is generated, the watchdog timer reset type signal WDTATRTYPE is set to high level and the watchdog timer stops operating.

The power-on-clear circuit holds the microcontroller in the reset state as long as the power supply voltage does not exceed the threshold level V_{POC}.

The following figure shows the POCRES generation timing.

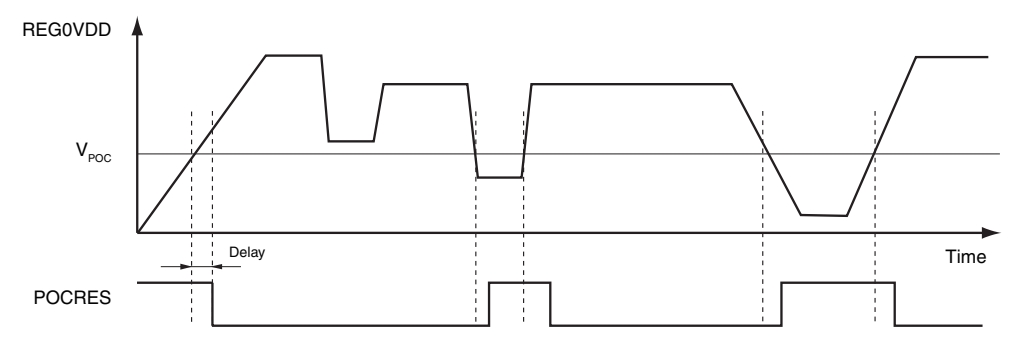


Figure 12-3 POC reset timing

Delay A delay occurs between when REG0VDD crosses the V_{POC} level and when POCRES is generated and canceled. For details about the delay time specifications, see the Data Sheet.

12.2.3 Low-voltage indicator (LVI)

The low voltage indicator (LVI) constantly compares the power supply voltage REG0VDD with the LVI internal reference voltage V_{LVI}.

If REG0VDD falls below the internal reference voltage (REG0VDD < V_{LVI}), the internal reset signal $\overline{\text{LVIRES}}$ and the system reset signal SYSRES are generated.

In addition, the $\overline{\text{LVIRES}}$ flag (RESF.RESF7 bit) is set. The RESF.RESF7 bit is not cleared automatically even if REG0VDD exceeds V_{LVI}. The RESF.RESF7 bit is cleared by

- setting the RESFC.RESFC7 bit to 1
- the occurrence of the power-on clear reset POCRES (for M1 products) or a reset triggered by $\overline{\text{RESET}}$ pin input (for M2 products)

Note Even if $\overline{\text{LVIRES}}$ is generated, the watchdog timer continues operating because the WDTATRTYP signal is not set to high level.

LVI reference voltage The LVI reference voltage V_{LVI} can be selected from three levels by using the LVICNT.LVICNT[2:0] bits.If the LVICNT.LVICNT[2:0] bits are set to 000_B, the LVI is disabled. For details about the specifications of the V_{LVI}, internal voltage reference levels, see *Table 12-5 "LVICNT register contents"*.

LVI interrupt The LVI interrupt INTLVI is asserted if:

- REG0VDD falls below the LVI reference voltage (REG0VDD < V_{LVI})
- REG0VDD rises above the LVI reference voltage (REG0VDD < V_{LVI})

The INTLVI interrupt can be used as a source for waking the system up from any standby mode. For details, see *Chapter 10 Standby Controller (STBC)*.

$\overline{\text{LVIRES}}$ mask Generating the $\overline{\text{LVIRES}}$ signal can be disabled as follows:

- LVICNT.LVIRESMK bit = 0: $\overline{\text{LVIRES}}$ is not masked (enabled).
- LVICNT.LVIRESMK bit = 1: $\overline{\text{LVIRES}}$ is masked (disabled).

The following figure shows the timing of the $\overline{\text{LVIRES}}$ signal and the RESF.RESF7 bit.

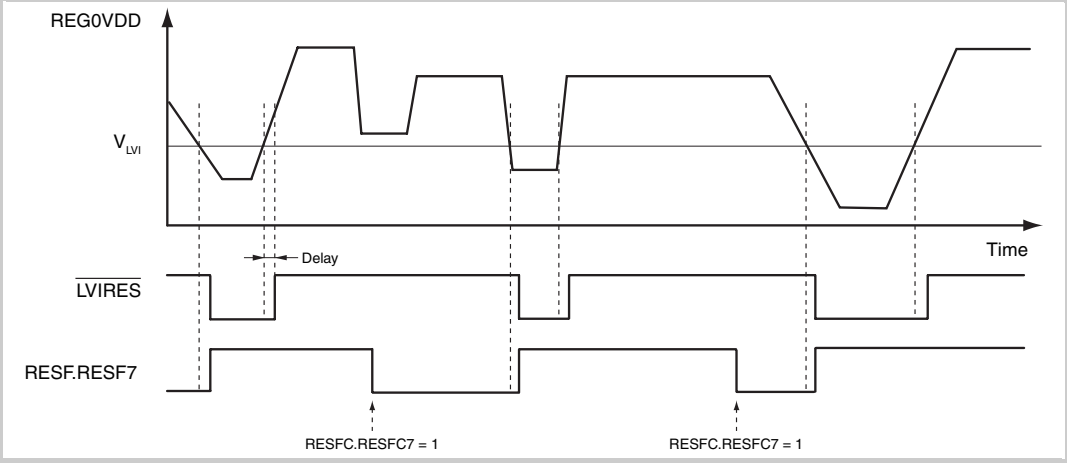


Figure 12-4 LVI reset timing

Delay After REG0VDD crosses the V_{LVI} level, a delay occurs between when $\overline{LVIRES}$ is asserted and the RESF.RESF7 bit is set. For details about the delay time specifications, see the Data Sheet.

12.2.4 RAM retention voltage indicator (RAMHF)

The RAM retention voltage indicator (RAMHF) constantly compares the power supply voltage REG0VDD with the RAMHF internal reference voltage V_{RAMHF}.

For details about the specifications of the V_{RAMHF} internal reference voltage level, see the Data Sheet.

<R> **BURAM contents retention**

<R> As long as the power supply voltage does not fall below V_{RAMHF}, the contents of the backup RAM (BURAM) are retained and therefore does not need to be restored. If REG0VDD falls below V_{RAMHF}, the BURAM contents are assumed to have altered. Therefore, the entire BURAM must be initialized before continuing operation.

<R> If REG0VDD falls below the internal reference voltage (REG0VDD < V_{RAMHF}), the VLVF.VLVF bit is set. The VLVF.VLVF bit is not cleared automatically even if

<R> REG0VDD exceeds V_{RAMHF}. The VLVF bit is cleared by

- setting the VLVFC bit to 1

The following figure shows the timing of setting the VLVF bit.

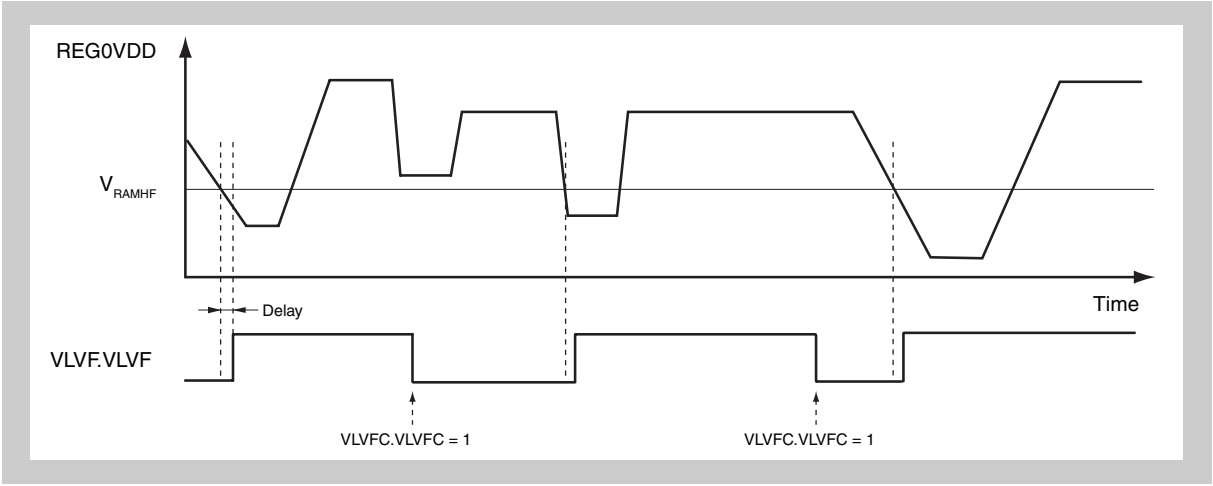


Figure 12-5 RAMHF reset timing

Delay A delay occurs between when REG0VDD crosses the V_{RAMHF} level and when the VLVF bit is set. For details about the delay time specifications, see the Data Sheet.

12.2.5 External $\overline{\text{RESET}}$

A reset is executed when a low-level signal is applied to the $\overline{\text{RESET}}$ pin.

In products that have a POC (for M1 products), the $\overline{\text{RESET}}$ flag (RESF.RESF8 bit) is set, and the system reset signal SYSRES and power-up reset signal PURES are generated. The RESF.RESF8 bit is not cleared automatically even if $\overline{\text{RESET}}$ is deasserted. The RESF.RESF8 bit is cleared by:

- setting the RESFC.RESFC8 bit to 1
- the generation of the power-on clear reset POCRES (for M1 products) or by the input of a reset signal via the $\overline{\text{RESET}}$ pin (for M2 products)

Clock generators

The clock generators are reset because the input of the external reset signal (RESET) asserts the power-up reset signal (PURES). For details, see *Chapter 9 "Clock Controller"*.

The $\overline{\text{RESET}}$ signal passes through an analog noise filter to prevent erroneous resets due to noise.

The figure below shows the timing of when an external reset occurs. This figure also shows the effect of the noise eliminator.

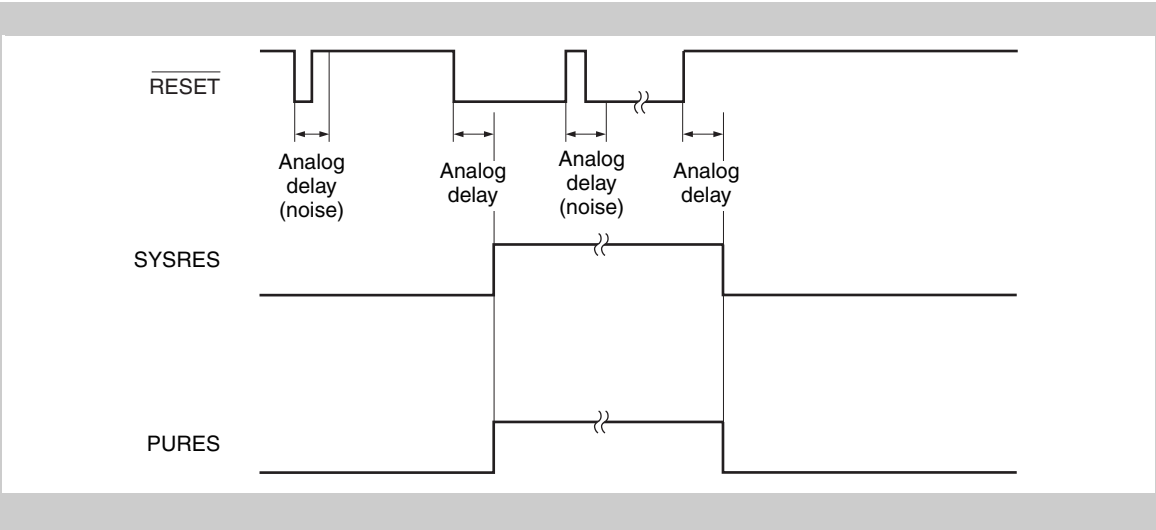


Figure 12-6 External $\overline{\text{RESET}}$ timing

The analog delay is caused by the analog filter. The filter regards pulses up to a certain width as noise and suppresses them. For the minimum $\overline{\text{RESET}}$ pulse width, see the Data Sheet.

12.2.6 Watchdog timer reset

The watchdog timer can be configured to generate a reset if the watchdog time expires. When a watchdog reset occurs, the watchdog timer reset flags (RESF.RESF1 for WDTA0RES and RESF.RESF2 for WDTA1RES) are set and the system reset signal SYSRES is generated.

The RESF.RESF1 (RESF.RESF2) bit is not cleared automatically even if WDTA0RES (WDTA1RES) is deasserted. It is cleared by

- setting the RESFC.RESFC1 (RESFC.RESFC2) bit to 1
- the generation of the power-on clear reset signal POCRES (for M1 products) or by the input of a reset via the $\overline{\text{RESET}}$ pin (for M2 products)

WDTATRTYP Whether the watchdog timer continues or stops operating when the WDTATRTYP signal is input depends on the reset source. For details, see Chapter *14 "Window Watchdog Timer A (WDTA)"*.

- When the reset source is POCRES (for M1 products) or $\overline{\text{RESET}}$ pin input (for M2 products), the WDTATRTYP signal is set to high level to sop the watchdog timer.
- Other reset sources set the WDTATRTYP signal to low level and the watchdog timer does not stop.

12.2.7 Software reset

The software reset SWRES can be generated by setting the SWRESA.SWRESA bit to 1.

This generates the system reset signal SYSRES and sets the reset flag RESF.RESF0 to 1. RESF.RESF0 is not cleared automatically. It is cleared by

- setting RESFC.RESFC0 to 1
- the generation of the power-on clear reset signal POCRES (for M1 products) or by the input of a reset via the $\overline{\text{RESET}}$ pin (for M2 products)

12.2.8 Clock monitor reset

The clock monitor can generate the following reset signals:

- $\overline{\text{CLMA0RES}}$, if a MainOsc failure is detected
- $\overline{\text{CLMA2RES}}$, if an 8 MHz IntOsc failure is detected
- $\overline{\text{CLMA3RES}}$, if a PLL0 failure is detected

Upon a clock monitor reset, the system reset signal SYSRES is generated and the respective reset flag in the RESF register is set. These flags are not cleared automatically. They are cleared by

- setting the RESFC.RESFC3 bit for $\overline{\text{CLMA0RES}}$, the RESFC.RESFC5 bit for $\overline{\text{CLMA2RES}}$, or the RESFC.RESFC6 bit for $\overline{\text{CLMA3RES}}$)
- the generation of the power-on clear reset signal POCRES (for M1 products) or by the input of a reset via the $\overline{\text{RESET}}$ pin (for M2 products)

12.2.9 Reset controller register protection

The following reset controller register is write-protected:

- Software reset register SWRESA

For details about writing to write-protected registers, see *3.10 “Write-Protected Registers”*.

12.3 Registers

This section describes all reset controller registers.

12.3.1 Reset controller register overview

The reset controller is controlled and operated by the following registers:

Table 12-1 Reset controller register overview

Register name	Symbol	Address
General reset flags registers		
Reset factor register	RESF	FF42 0160$_H$
Reset factor clear register	RESFC	FF42 0168$_H$
Software reset control register		
Software reset register	SWRESA	FF42 0204$_H$
Low voltage indicator reset control registers		
LVI control register	LVICNT	FF42 0200$_H$
RAM retention voltage flag control registers		
RAM retention voltage detection flag register	VLVF	FF42 0180$_H$
RAM retention voltage detection flag clear register	VLVFC	FF42 0188$_H$
Protection command registers		
Protection command register 2	PROTCMD2	FF42 0300$_H$
Protection status register 2	PROTS2	FF42 0304$_H$

12.3.2 General reset flag register details

(1) RESF- Reset factor register

This register contains information about which type of resets occurred since the last power-on clear reset.

The corresponding flag in this register is set according to each reset condition. For example, if the clock monitor reset $\overline{\text{CLMA0RES}}$ occurs after the watchdog timer reset WDTA0RES, 0000 000A$_H$ is read from this register.

Access This register is read-only, in 32-bit units.

Address FF42 0160$_H$

Initial value 0000 0000$_H$. This register is initialized by the power-on-clear reset POCRES (for M1 products), or the input of a reset via the $\overline{\text{RESET}}$ pin (for M2 products).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	RESF8[a]	RESF7	RESF6	RESF5	0	RESF3	RESF2	RESF1	RESF0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

a) Clear this bit to 0 for M2 products.

Table 12-2 RESF register contents

Bit position	Bit name	Function
8	RESF8	External reset flag (for M1 products) 0: No $\overline{\text{RESET}}$ reset occurred 1: $\overline{\text{RESET}}$ reset has occurred
7	RESF7	Low voltage indicator reset flag 0: No $\overline{\text{LVIRES}}$ reset occurred 1: $\overline{\text{LVIRES}}$ reset has occurred
6	RESF6	Clock monitor CLMA3 reset flag 0: No $\overline{\text{CLMA3RES}}$ reset occurred 1: $\overline{\text{CLMA3RES}}$ reset has occurred
5	RESF5	Clock monitor CLMA2 reset flag 0: No $\overline{\text{CLMA2RES}}$ reset occurred 1: $\overline{\text{CLMA2RES}}$ reset has occurred
3	RESF3	Clock monitor CLMA0 reset flag 0: No $\overline{\text{CLMA0RES}}$ reset occurred 1: $\overline{\text{CLMA0RES}}$ reset has occurred
2	RESF2	Watchdog timer WDTA1 reset flag 0: No WDTA1RES reset occurred 1: WDTA1RES reset has occurred
1	RESF1	Watchdog timer WDTA0 reset flag 0: No WDTA0RES reset occurred 1: WDTA0RES reset has occurred
0	RESF0	Software reset flag 0: No SWRES reset occurred 1: SWRES reset has occurred

(2) RESFC - Reset factor clear register

This register is used to clear the reset flags of the RESF register.

Access This register can be read or written in 32-bit units.

Address FF42 0168$_H$

Initial value Reading this register always returns 0000 0000$_H$. This register is initialized by the power-on-clear reset POCRES (for M1 products), or the input of a reset via the $\overline{\text{RESET}}$ pin (for M2 products).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	RESFC8[a]	RESFC7	RESFC6	RESFC5	0	RESFC3	RESFC2	RESFC1	RESFC0
R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W

a) Clear this bit to 0 for M2 products.

Table 12-3 RESFC register contents

Bit position	Bit name	Function
8	RESFC8	Clears the external reset flag RESF.RESF8 (for M1 products) 0: Does not function 1: Clears RESF.RESF8.
7	RESFC7	Clears the low voltage indicator reset flag RESF.RESF7 0: Does not function 1: Clear RESF.RESF7.
6	RESFC6	Clears the clock monitor CLMA3 reset flag RESF.RESF6. 0: Does not function 1: Clears RESF.RESF6
5	RESFC5	Clears the clock monitor CLMA2 reset flag RESF.RESF5. 0: Does not function 1: Clears RESF.RESF5
3	RESFC3	Clears the clock monitor CLMA0 reset flag RESF.RESF3. 0: Does not function 1: Clear RESF.RESF3
2	RESFC2	Clears the watchdog timer WDTA1 reset flag RESF.RESF2. 0: Does not function 1: Clear RESF.RESF2
1	RESFC1	Clears the watchdog timer WDTA0 reset flag RESF.RESF1 flag. 0: Does not function 1: Clear RESF.RESF1
0	RESFC0	Clears the software reset flag RESF.RESF0. 0: Does not function 1: Clear RESF.RESF0

12.3.3 Software reset control register details

(1) SWRESA - Software reset register

This register is used to generate the software reset SWRES.

Access This register can be read or written in 32-bit units.

This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register PROTCMD2.
For details, see *12.2.9 "Reset controller register protection"*.

Address FF42 0204$_H$

Initial value Reading this register always returns 0000 0000$_H$.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SWRESA
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W

Table 12-4 SWRESA register contents

Bit position	Bit name	Function
0	SWRESA	Software reset control 0: Does not function 1: Generates the software reset SWRES.

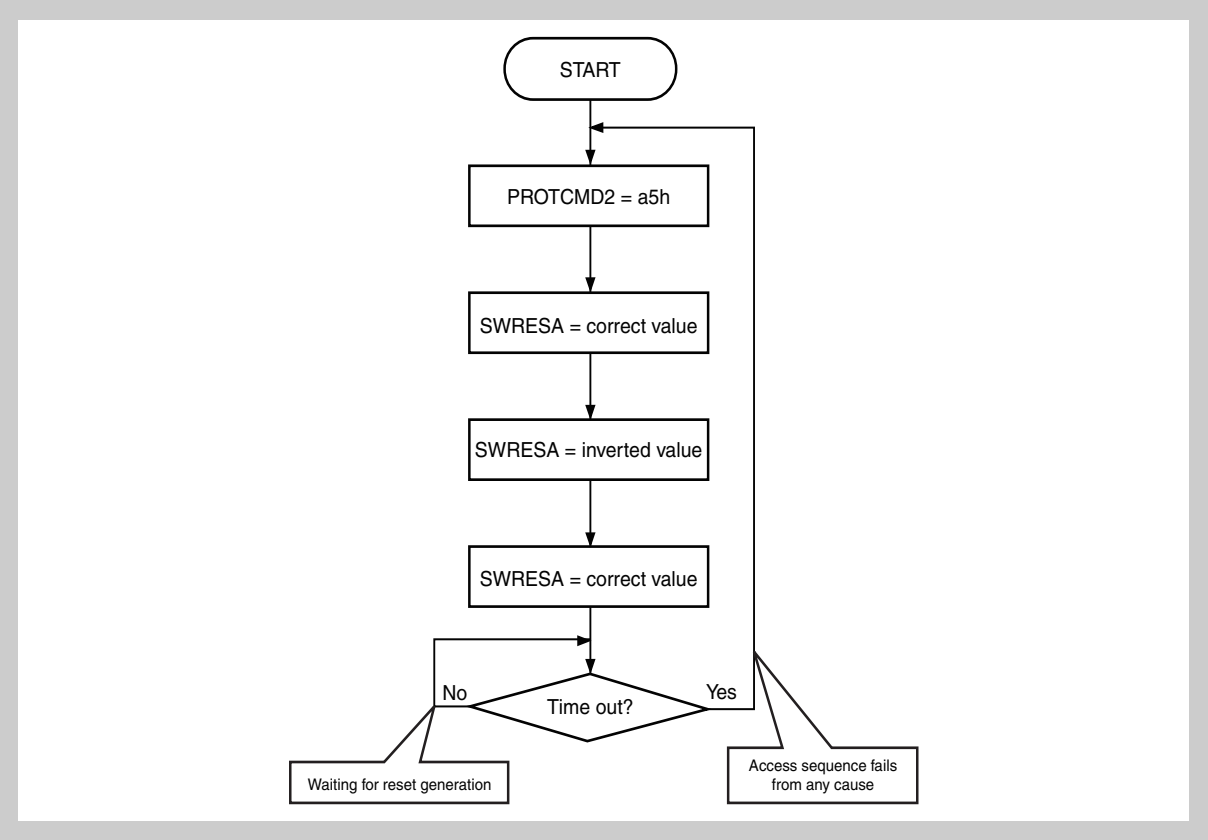


Figure 12-7 Software reset control method

12.3.4 Low voltage indicator reset control registers

(1) LVICNT - LVI control register

This register is used to control the low voltage indicator and to select the LVI detection level.

Access This register can be read or written in 32-bit units.

This register is write-protected, and can only be written by using a special instruction sequence, initiated by using the protection command register PROTCMD2.
For details, see *12.2.9 "Reset controller register protection"*.

Address FF42 0200_H

Initial value $0000\ 0000_H$. This register is initialized by the power-on-clear reset POCRES (for M1 products), or the input of a reset via the $\overline{\text{RESET}}$ pin (for M2 products).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	LVIRESMK	LVICNT[2:0]		
R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W

Table 12-5 LVICNT register contents

Bit position	Bit name	Function
3	LVIRESMK	LVI reset $\overline{\text{LVIRES}}$ mask 0: $\overline{\text{LVIRES}}$ unmasked 1: $\overline{\text{LVIRES}}$ masked (generating $\overline{\text{LVIRES}}$ is disabled)
2 to 0	LVICNT[2:0]	LVI detection level

<R>

LVICNT[2:0]	Detection level	
	M1 products	M2 products
000_B	LVI is deactivated	
100_B	LVI level 1 (3.1 V ±0.1 V)	
101_B	Setting prohibited	LVI level 2 (2.9 V ±0.1 V)
$11X_B$	Setting prohibited	
Other than above	Setting prohibited	

For details about the LVI detection level specifications, see the *Data Sheet*.

Note If the selected LVI detection level is close to the power-on-clear detection level, both the POC circuit and the LVI circuit might detect a low voltage at the same time.In this case, the power-on-clear reset POCRES takes effect and the reset factor register RESF is cleared. Therefore, the LVI reset flag RESF.RESF7 does not indicate LVI detection.

12.3.5 RAM retention voltage detection flag control registers

(1) VLVF - RAM retention voltage detection flag register

This register indicates the VLVF detection status.

Access This register is read-only, in 32-bit units.

Address FF42 0180$_H$

<R> **Initial value** 0000 0001$_H$. This register is not initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	VLVF
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 12-6 VLVF register contents

Bit position	Bit name	Function
0	VLVF	RAM retention voltage detection status 0: RAM retention voltage or lower not detected 1: RAM retention voltage or lower detected

<R>
<R>

(2) VLVFC - RAM retention voltage detection flag clear register

This register is used to clear the VLVF.VLVF bit.

Access This register can be read or written in 32-bit units.

Address FF42 0188$_H$

Initial value Reading this register always returns 0000 0000$_H$.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	VLVFC
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W

Table 12-7 VLVFC register contents

Bit position	Bit name	Function
0	VLVFC	Clear the VLVF.VLVF bit. 0: Does not function 1: Clears the VLVF.VLVF bit.

12.3.6 Protection command register details

(1) PROTCMDm – Protection command register m (m = 2)

This register is used to initiate the write protection unlock sequence.

Access This register is write-only, in 8-bit units.

(This register is always read as 0.)

Address PROTCMD2: FF42 0300_H

Initial value 00_H. This register is initialized by any reset.

<R>

7	6	5	4	3	2	1	0
PCMD7	PCMD6	PCMD5	PCMD4	PCMD3	PCMD2	PCMD1	PCMD0
W	W	W	W	W	W	W	W

For details about the usage of this register, see *3.10.6 (1) "PROTCMDm – Protection command register m (m = 2)"*.

Table 12-8 PROTCMDm register contents

	Bit position	Bit name	Function
<R>	7 to 0	PCMD[7:0]	Indicates the protection command to enable writing to write-protected registers.

(2) PROTSm – Protection status register m (m = 2)

This register shows the status of the protection unlock sequence initiated by PROTCMDm.

Access This register is read-only, in 8-bit units.

Writing to this register is ignored.

Address PROTS2: FF42 0304_H

Initial value 00_H. This register is initialized by a reset, or writing A5H to the PROTCMDm register.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PROTERR
R	R	R	R	R	R	R	R

Table 12-9 PROTSm register contents

Bit position	Bit name	Function
0	PROTERR	Protected write sequence error monitor 0: No protection error 1: Protection error occurred

Chapter 13 OS Timer (OSTM)

This chapter contains a generic description of the OS timer (OSTM).

The first section describes all properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names.
The subsequent sections describe the features that apply to all implementations.

13.1 V850E2/Sx4-H OSTM Features

Instances

This microcontroller has the following number of instances of OSTM:

Table 13-1 Instances of OSTM

OSTM	
Number of instances	1
Names	OSTM0

Instances index n

Throughout this chapter, the individual instances of OSTM are identified by the index “n” (n = 0), for example, OSTMnCMP for the OSTM compare register.

Register addresses

All OSTM register addresses are given as addresses offset from the individual base address <OSTMn_base>.
The <OSTMn_base> addresses of each OSTMn are listed in the following table:

Table 13-2 Register base addresses <OSTMn_base>

OSTMn	<OSTMn_base> address
OSTM0	FF80 0000_H

Clock supply

The following clock is supplied to OSTM:

Table 13-3 OSTM clock supply

OSTMn	Clock	Connected to:
OSTM0	PCLK	Clock generator PCLK

Interrupts

OSTM can generate the following interrupt requests:

Table 13-4 OSTMn interrupt requests

OSTMn signals	Function	Connected to:
OSTM0TINT	OSTMn interrupt	Interrupt Controller INTOSTM0I

OSTM hardware reset

OSTM and its registers are initialized by the following reset signal:

Table 13-5 OSTMn reset signal

OSTMn	Reset signal
OSTM0	System reset SYSRES

I/O signals

The I/O signals of OSTM are listed in the following table:

Table 13-6 OSTMn I/O signals

OSTMn signals	Function	Connected to:
OSTMnTCKE	Count clock enable	Fixed to 1.
OSTMnTSST	Count start	Fixed to 0.
OSTMnTOUT	OS Timer output	No connection

13.2 Functional Overview

Features summary

The OS timer has the following features:

- Two operation modes
  - Interval timer mode
  - Free-run compare mode
- Two output modes (if OSTMnTTOUT signal is externally output)
  - Software control mode
  - Timer output toggle mode
- Synchronization with other peripheral functions (if a signal is input to OSTMnTSST)
- OSTMnTINT interrupt

The following block diagram shows the main components of the OS timer.

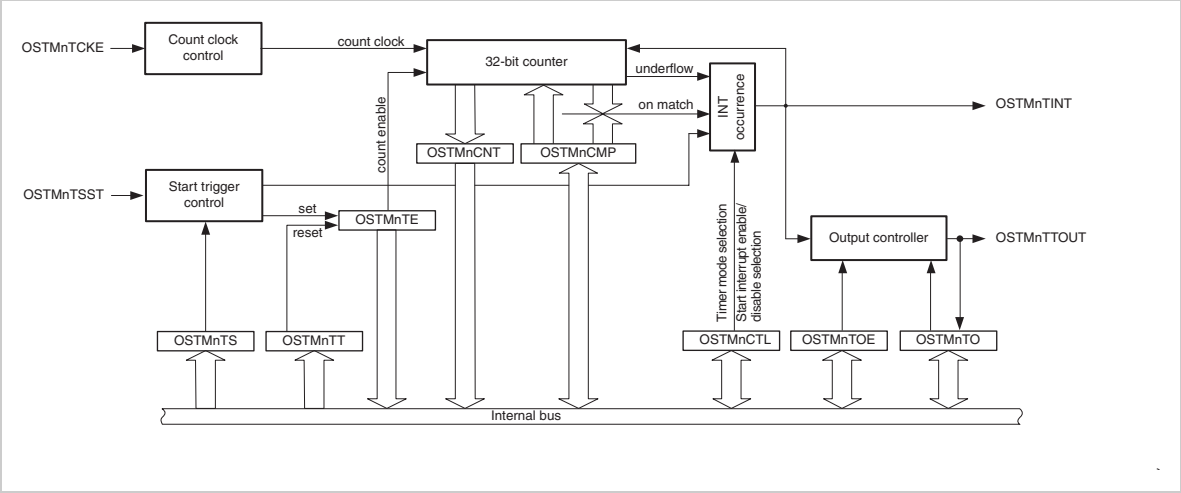


Figure 13-1 Block diagram of the OS timer

13.3 Functional Description

The OS timer is a 32-bit timer/counter.

It can be used as an interval timer or in free-run compare mode. The selected operation mode specifies the count direction (up/down) and controls the interrupt request generation.

The OS timer can be synchronized with other peripheral functions by using the count clock enable signal OSTMnTCKE and the count start signal OSTMnTSST. (See *13.3.1 "Count clock" on page 563* and *13.3.4 "Starting and stopping the timer" on page 565* for details.)

13.3.1 Count clock

The count clock of the OS timer is defined by PCLK and the OSTMnTCKE input:

- To use PCLK as the count clock, OSTMnTCKE must be fixed to 1.
- If the OSTMnTCKE signal is input, the count operation is based on this signal.

This is illustrated in the following figures.

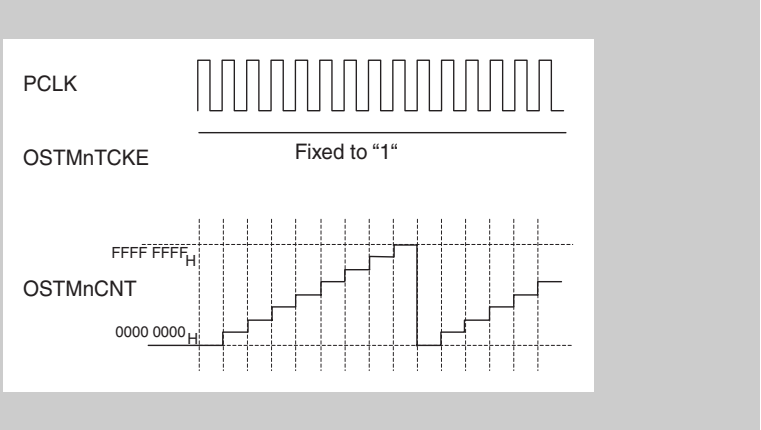


Figure 13-2 Count operation with OSTMnTCKE fixed to 1

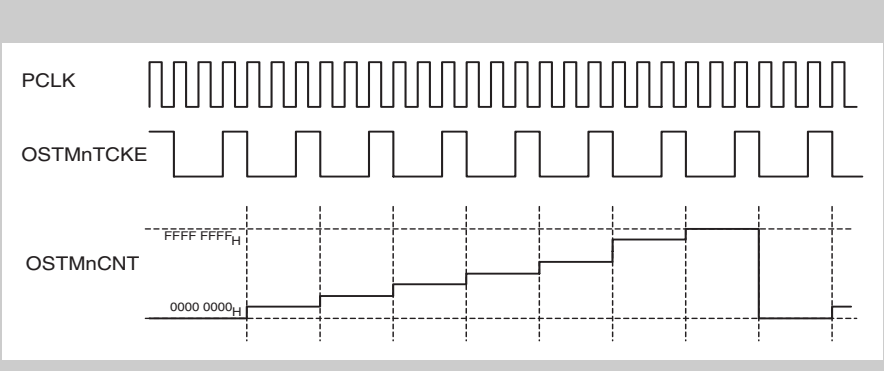


Figure 13-3 Count operation based on the OSTMnTCKE signal

13.3.2 Output modes

The OS timer has the following output modes:

- Software control mode:

  The level set to OSTMnTO.OSTMnTO is output to OSTMnTTOUT.

- Timer output toggle mode:

  OSTMnTTOUT toggles when the OSTMnTINT request is generated.

The output mode is selected by bit OSTMnTOE.OSTMnTOE.

Both output modes are illustrated in the following figure.

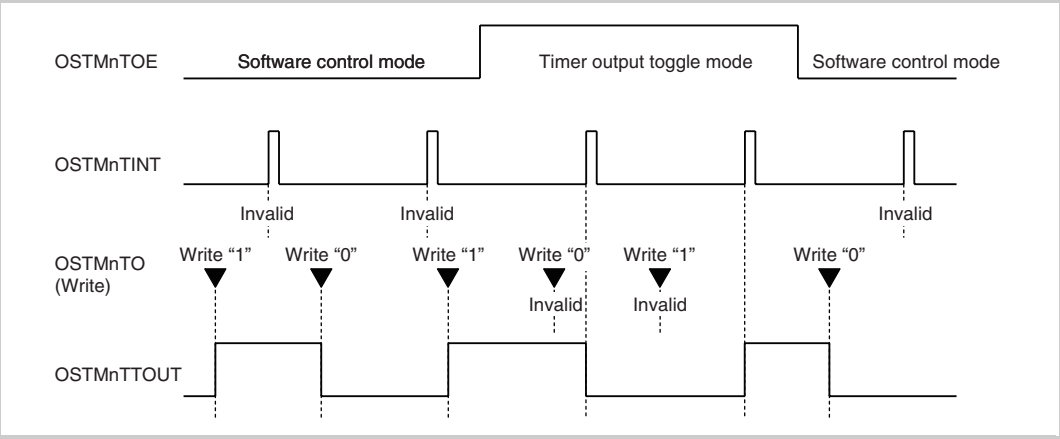


Figure 13-4 Timing diagram of output modes

The timing diagram above shows the following:

- In software control mode, OSTMnTTOUT changes to the value that was set in OSTMnTO.OSTMnTO.
- In timer output toggle mode, OSTMnTO.OSTMnTO and OSTMnTTOUT toggle when the OSTMnTINT interrupt request is generated.

13.3.3 Interrupt request generation

Interrupt request OSTMnTINT is generated on counter underflow (interval timer mode) or when the counter matches the compare value (free-run compare mode).

Additionally, an interrupt request can be generated at counter start or counter restart. This is controlled by bit OSTMnCTL.OSTMnMD0.

Since OSTMnTINT triggers toggling of OSTMnTTOUT in timer output toggle mode (OSTMnTOE.OSTMnTOE = 1), the setting of the OSTMnCTL.OSTMnMD0 bit also affects the output of OSTMnTTOUT.

This is illustrated in the following figure.

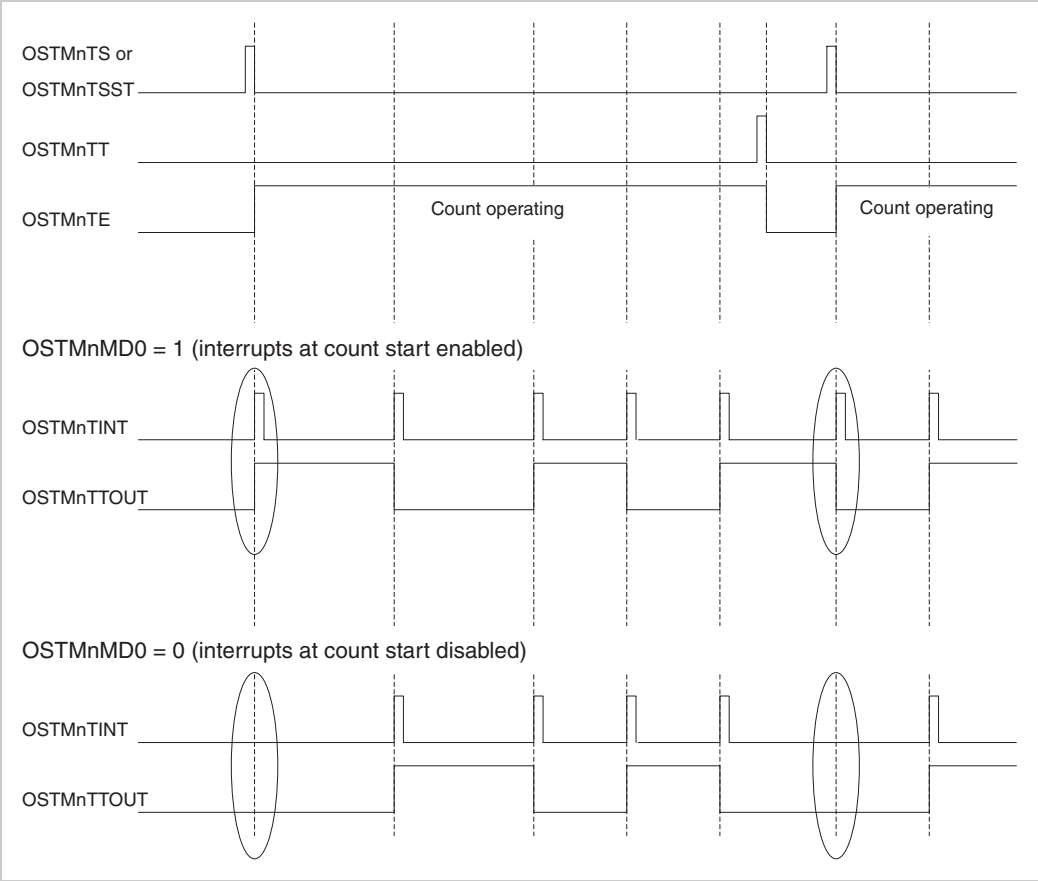


Figure 13-5 Interrupt generation at count start

13.3.4 Starting and stopping the timer

The OS timer is started and stopped as follows:

Start The timer is started

- by setting OSTMnTS.OSTMnTS, or
- when the OSTMnTSST signal changes from 0 to 1.

The status bit OSTMnTE.OSTMnTE is set to 1.

Depending on the operation mode, the counter starts to count down or to count up. See *13.3.5 “Interval timer mode” on page 566* and *13.3.6 “Free-run compare mode” on page 569* for details.

Stop The timer is stopped by setting the bit OSTMnTT.OSTMnTT = 1.

Status bit OSTMnTE.OSTMnTE is cleared.

When the counter is stopped, the registers OSTMnTO and OSTMnCNT hold their current values until a new count operation starts.

Simultaneous start If the OSTMnTSST signal is connected to another internal peripheral function, the OS timer can be started in synchronization with the peripheral function.

13.3.5 Interval timer mode

In interval timer mode, the OSTM can be used as a reference timer generating interrupt requests at fixed intervals.

(1) Basic operation in interval timer mode

In interval timer mode, the timer counts down, starting from the value specified in the OSTMnCMP register. When the counter underflows (0000 0000_H is reached), an interrupt request OSTMnTINT is generated.

When using the interval timer mode, clear the OSTMnCTL.OSTMnMD1 bit to 0.

The OSTMnCMP register can be rewritten at any time. If it is rewritten during count operation, the counter loads the new OSTMnCMP value when the next 0000 0000_H is reached. Then the counter continues with the new value.

OSTMnTINT and OSTMnTTOUT periods

The periods of OSTMnTINT and OSTMnTTOUT are:

- OSTMnTINT occurrence period = count clock period * (OSTMnCMP + 1)
- OSTMnTTOUT period = OSTMnTINT occurrence period * 2

The following figure shows the basic operation of the OS timer in interval timer mode with counter start interrupt enabled (OSTMnCTL.OSTMnMD0 = 1) and OSTMnTTOUT in timer output toggle mode (OSTMnTOE.OSTMnTOE = 1):

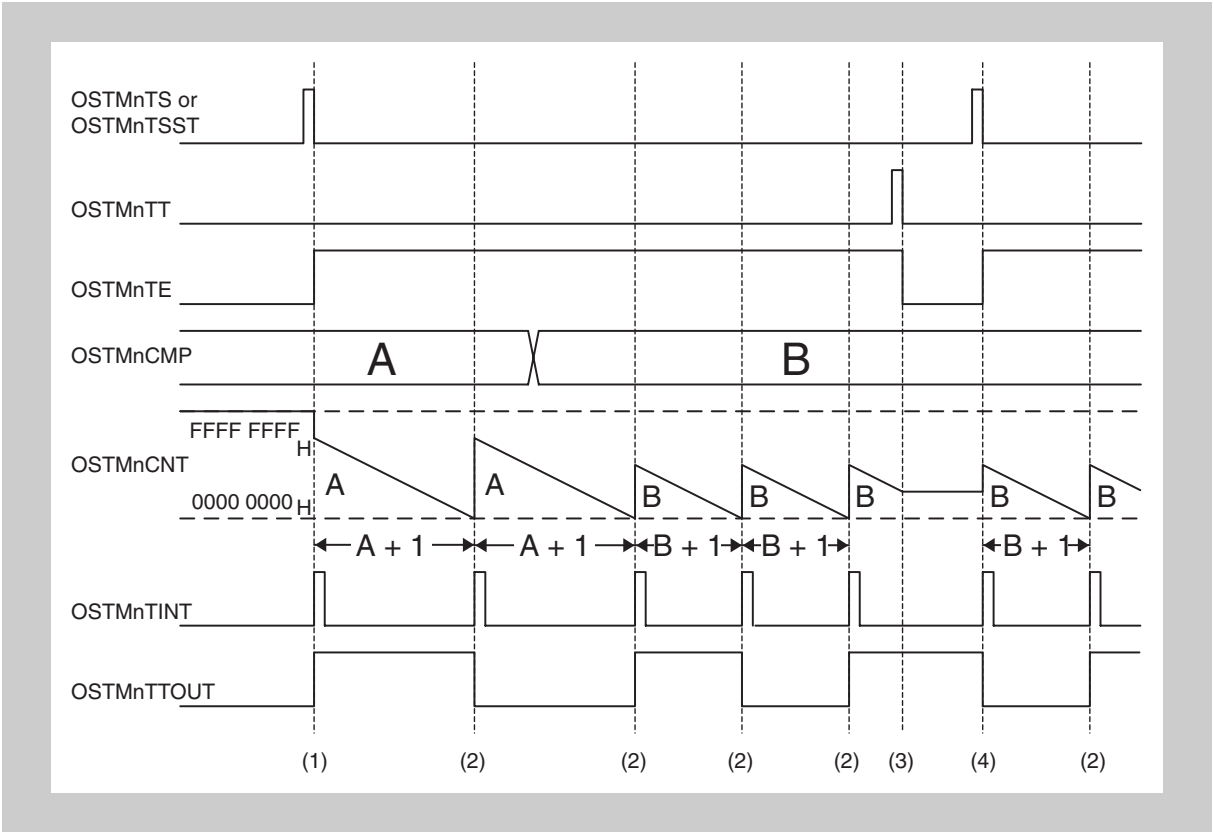


Figure 13-6 Timing diagram of OSTM in interval timer mode

The timing diagram above shows the following:

1. The counter starts counting when OSTMnTS.OSTMnTS = 1 or OSTMnTSST = 1. The OSTMnTE.OSTMnTE bit is set, indicating that a count operation is in progress.
   The counter starts counting down from the value of OSTMnCMP.
   If OSTMnCTL.OSTMnMD0 = 1, the interrupt request OSTMnTINT is generated when counting starts, and the OSTMnTTOUT output toggles.
   The counter value is indicated by the OSTMnCNT register.
2. When the counter reaches 0000 0000_H, the interrupt request OSTMnTINT is output and the OSTMnTTOUT output toggles. The counter loads the new start value from OSTMnCMP and continues to count down.
3. If OSTMnTT.OSTMnTT is set and the counter stops, the OSTMnTE.OSTMnTE bit is cleared, indicating that the counter is stopped.

   The counter holds the current value until it restarts counting.
4. If OSTMnTS.OSTMnTS or OSTMnTSST is set to restart counting, the counter loads the value from OSTMnCMP and starts counting down.

Forced restart

A forced restart of the counter is performed by setting OSTMnTS.OSTMnTS = 1 or generating a high to low transition of the OSTMnTSST signal during the count operation.

The counter loads the start value from the OSTMnCMP register and continues to count down.

The following figure shows the forced restart of the OS timer in interval timer mode, with counter start interrupt enabled (OSTMnCTL.OSTMnMD0 = 1) and OSTMnTTOUT in timer output toggle mode (OSTMnTOE.OSTMnTOE = 1):

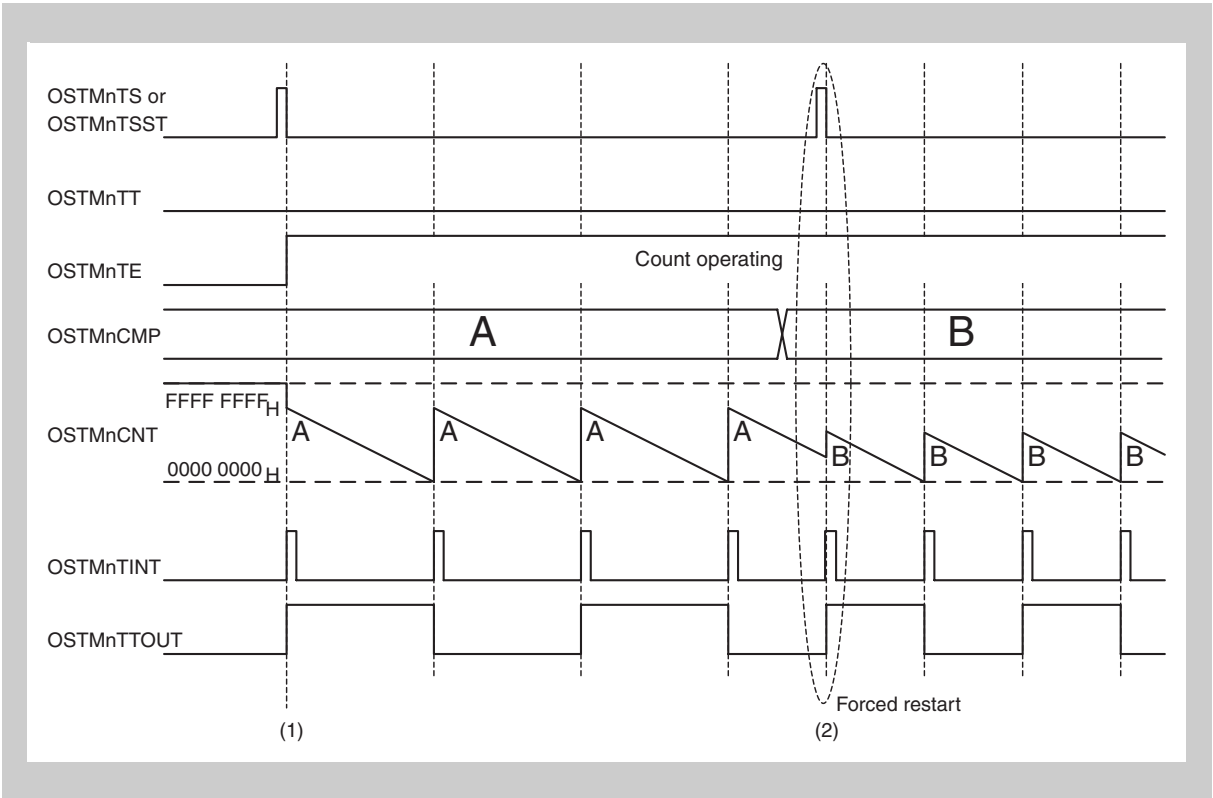


Figure 13-7 Timing diagram of forced restart in interval timer mode

The timing diagram above shows the following:

1. The counter is started as shown in *Figure 13-6 "Timing diagram of OSTM in interval timer mode" on page 566*.
2. If OSTMnTS.OSTMnTS or OSTMnTSST is set while the counter is operating (OSTMnTE.OSTMnTE = 1), the counter restarts counting.

   The counter immediately restarts counting down, starting with the current value of OSTMnCMP.
   If OSTMnCTL.OSTMnMD0 = 1, the interrupt request OSTMnTINT is generated when counting starts, and the OSTMnTTOUT output toggles.

(2) Operation when OSTMnCMP = 0000 0000$_H$

When OSTMnCMP = 0000 0000$_H$ the OS timer behaves as follows:

- While the counter is enabled, the OSTMnTINT interrupt request is always 1.
- If OSTMnTTOUT output is in timer output toggle mode, OSTMnTTOUT toggles at each count clock cycle.

The following figure shows the operation of the OS timer, when OSTMnCMP = 0000 0000$_H$, counter start interrupt is enabled (OSTMnCTL.OSTMnMD0 = 1) and OSTMnTTOUT is in timer output toggle mode (OSTMnTOE.OSTMnTOE = 1).

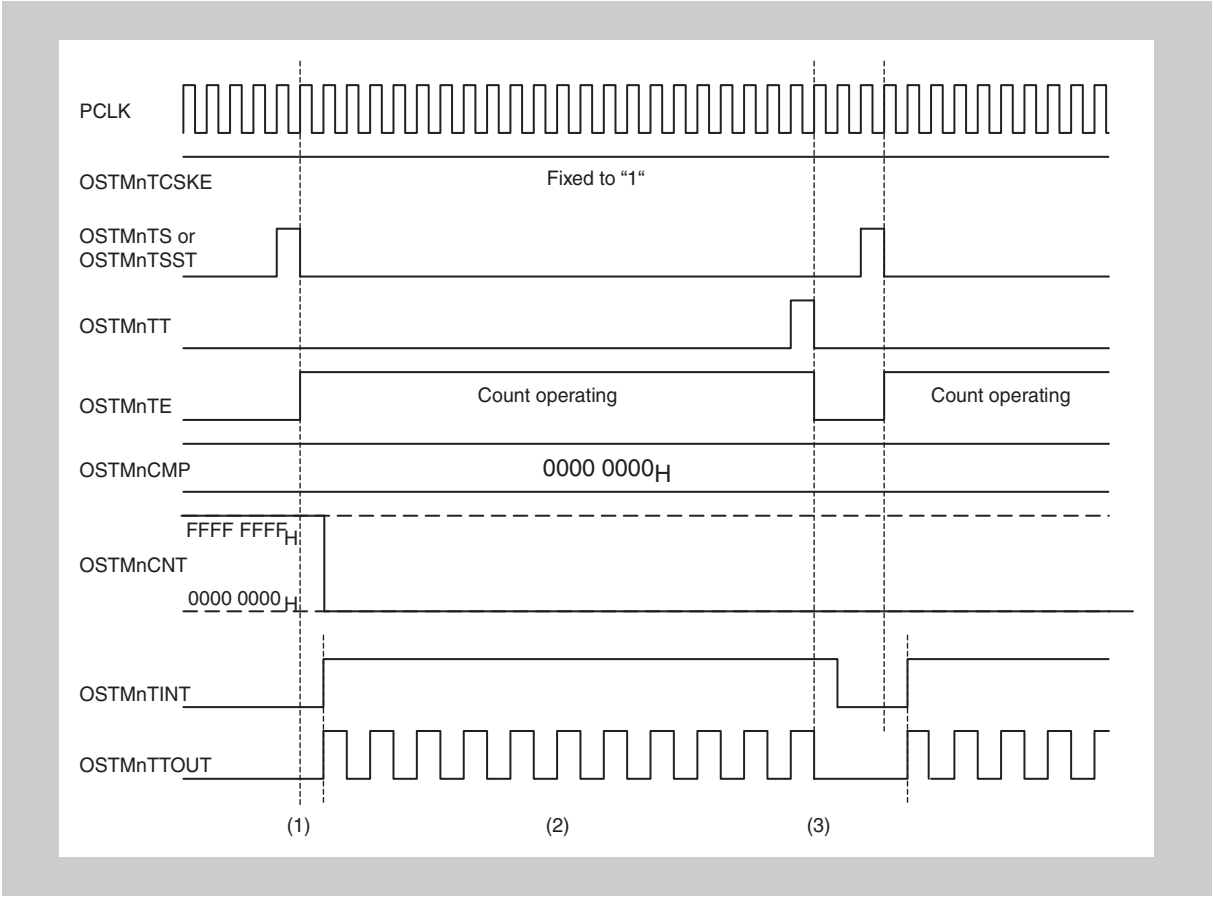


Figure 13-8 Timing diagram when OSTMnCMP = 0000 0000$_H$ in interval timer mode

The timing diagram above shows the following:

1. After counter start, the counter starts counting, but will be loaded with the OSTMnCMP value, and thus remains 0000 0000_H.
2. The interrupt request OSTMnTINT is continuously asserted and OSTMnTTOUT starts toggling at each count clock cycle.
3. After counter stop, the interrupt request OSTMnTINT is deasserted and OSTMnTTOUT stops toggling.

(3) Initialization for interval timer mode

The setting procedure in interval timer mode after a reset release is described below:

Initialization

1. Set the start value of the down-counter in the OSTMnCMP register.
2. To use the OSTMnTTOUT output pin:
   - In software control mode, initialize OSTMnTO
   - Select the output mode (OSTMnTOE.OSTMnTOE = 1).
3. Select the interval timer mode by clearing bit OSTMnCTL.OSTMnMD1.
4. Select the interrupt mode at counter start (OSTMnCTL.OSTMnMD0 = 1).

13.3.6 Free-run compare mode

(1) Basic operation in free-run compare mode

In free-run compare mode, the counter counts up from 0000 0000_H to FFFF $FFFF_H$. When the value of the OSTMnCMP register matches the current count value, the OSTMnTINT interrupt request is output.

The free-run compare mode is selected by setting OSTMnCTL.OSTMnMD1 = 1.

The OSTMnCMP register can be rewritten at any time.

The following figure shows the basic operation of the OS timer in free-run compare mode with counter start enabled (OSTMnCTL.OSTMnMD0 = 1) and OSTMnTTOUT in timer output toggle mode (OSTMnTOE.OSTMnTOE = 1):

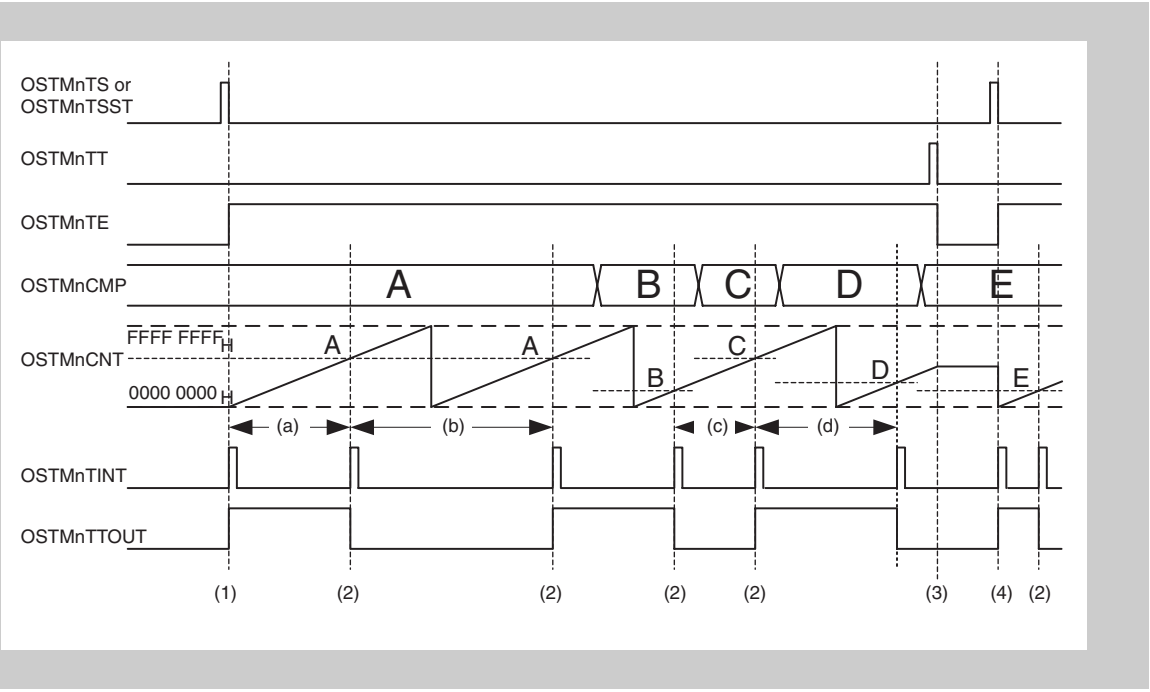


Figure 13-9 Timing diagram of OSTM in free-run compare mode

The timing diagram above shows the following:

1. The counter starts counting when OSTMnTS.OSTMnTS = 1 or OSTMnTSST = 1. The OSTMnTE.OSTMnTE bit is set, indicating that the counter is operating.
   The counter counts up from 0000 0000_H to FFFF $FFFF_H$. The counter value is indicated by OSTMnCNT register.
2. When the value of the OSTMnCMP register matches the current counter value, the interrupt request OSTMnTINT is output and the OSTMnTTOUT output toggles.
3. At counter stop (OSTMnTT.OSTMnTT = 1), the OSTMnTE.OSTMnTE bit is cleared, indicating that the counter is disabled.

   The counter holds the current value until it restarts counting.
4. If OSTMnTS.OSTMnTS or OSTMnTSST is set to restart counting, the counter starts counting from 0000 0000_H.

The OSTMnTINT occurrence period is different at count start and depends on the old and new compare value if OSTMnCMP is rewritten during operation:

Table 13-7 OSTMnTINT occurrence timing

Old compare	New compare	Counter value at time of rewrite	OSTMnTINT occurrence period	Label in timing diagram
Counter start			(A + 1) * count clock period	(a)
A	A	No rewrite	(FFFF FFFF$_H$ + 1) * count clock period	(b)
B	C > B	B < counter value < C	(C - B) * count clock period	(c)
C	D < C	Counter value > D, C	(FFFF FFFF$_H$ - C + D + 1) * count clock period	(d)

Forced restart A forced restart operation is not performed even if the bit OSTMnTS.OSTMnTS is set or OSTMnTSST = 1 during the count operation. The counter ignores this setting and continues counting.

(2) Operation when OSTMnCMP = 0000 0000$_H$

The following figure shows the operation of the OS timer when OSTMnCMP = 0000 0000$_H$, counter start interrupt is enabled (OSTMnCTL.OSTMnMD0 = 1) and OSTMnTTOUT is in timer output toggle mode (OSTMnTOE.OSTMnTOE = 1).

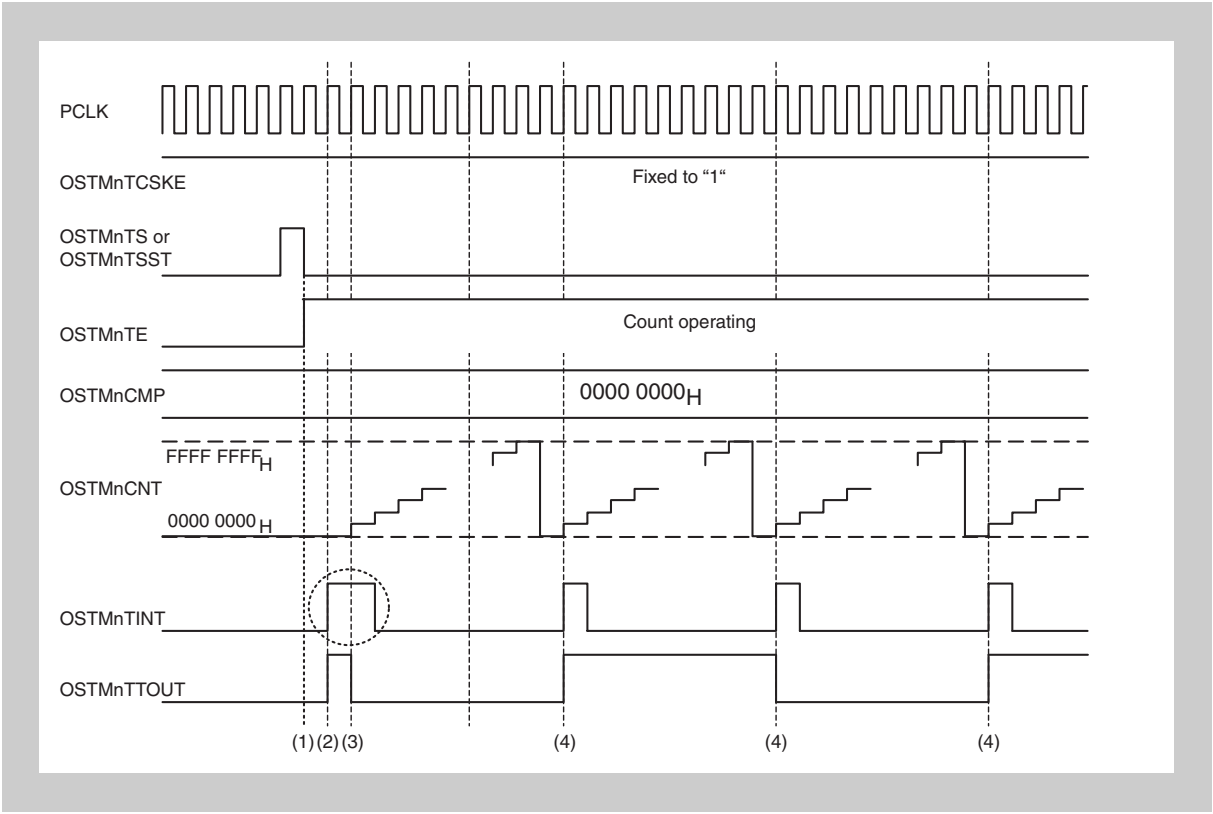


Figure 13-10 Timing diagram when OSTMnCMP = 0000 0000$_H$ in free-run compare mode

The timing diagram above shows the following:

1. After counter start, the counter starts counting up from 0000 0000_H to FFFF $FFFF_H$.
2. The interrupt request OSTMnTINT at counter start is generated.
3. If the current count value matches OSTMnCMP, the compare interrupt is generated. In the above case with OSTMnCMP = 0000 0000_H, OSTMnTINT stays active for 2 PCLK cycles.
4. Every FFFF $FFFF_H$ clock cycles the interrupt request OSTMnTINT is output and OSTMnTTOUT toggles.

(3) Initialization for free-run compare mode

The setting procedure in free-run compare mode after a reset release is described below:

Initialization

1. Set the compare value in the OSTMnCMP register.
2. To use the OSTMnTTOUT output pin:
   - In software control mode, initialize OSTMnTO.
   - Select the output mode (OSTMnTOE.OSTMnTOE = 1).
3. Select the free-run compare mode by setting the bit OSTMnCTL.OSTMnMD1.
4. Select the interrupt mode at counter start by the bit OSTMnCTL.OSTMnMD0.

13.4 Registers

This section contains a description of all registers of the OS timer.

13.4.1 OS timer register overview

The OS timer is controlled and operated by the following registers:

Table 13-8 OS timer register overview

Register name	Symbol	Address
OSTM compare register	OSTMnCMP	<OSTMn_base>
OSTM counter register	OSTMnCNT	<OSTMn_base> + 4_H
OSTM output register	OSTMnTO	<OSTMn_base> + 8_H
OSTM output enable register	OSTMnTOE	<OSTMn_base> + C_H
OSTM count enable status register	OSTMnTE	<OSTMn_base> + 10_H
OSTM count start trigger register	OSTMnTS	<OSTMn_base> + 14_H
OSTM count stop trigger register	OSTMnTT	<OSTMn_base> + 18_H
OSTM control register	OSTMnCTL	<OSTMn_base> + 20_H

13.4.2 OS timer register details

(1) OSTMnCMP - OSTM compare register

This register stores the start value of the down-counter or the value with which the counter is compared, depending on the operation mode.

Access This register can be read/written in 32-bit units.

Address <OSTMn_base>

Initial Value 0000 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSTMnCMP[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSTMnCMP[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 13-9 OSTMnCMP register contents

Bit position	Bit name	Function
31 to 0	OSTMnCMP[31:0]	• In interval timer mode: start value of the down-counter • In free-run compare mode: compare value

(2) OSTMnCNT - OSTM counter register

This register indicates the count value of the timer.

<R> This register can be written while the counter is stopped (OSTM0TE = 0).

Access This register can be read in 32-bit units.

Address <OSTMn_base> + 4_H

Initial Value The initial value depends on the operation mode of the OS timer, see *Table 13-11 "Correlation between operation mode, counting direction and initial value" on page 574*. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSTMnCNT[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSTMnCNT[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 13-10 OSTMnCNT register contents

Bit position	Bit name	Function
31 to 0	OSTMnCNT[31:0]	32-bit counter value

The following table shows the correlation between operation mode, counting direction and initial value. The initial value is the value that is read after the operating mode has been changed.

Table 13-11 Correlation between operation mode, counting direction and initial value

Timer operation mode	OSTMnCTL.OSTMnMD1	Counting direction	Initial value
Interval Timer Mode	0[a]	Down	FFFF FFFF$_H$
Free Running Compare Mode	1	Up	0000 0000$_H$

a) Value after reset.

(3) OSTMnTO - OSTM output register

This register specifies and reads the level of OSTMnTTOUT.

Access This register can be read/written in 8-bit units. It can only be written when the software control mode is enabled (OSTMnTOE.OSTMnTOE = 0).

Address <OSTMn_base> + 8_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OSTMnTO
R	R	R	R	R	R	R	R/W

Table 13-12 OSTMnTO register contents

Bit position	Bit name	Function
0	OSTMnTO	Specifies/reads the level of OSTMnTTOUT: 0: Low level 1: High level

(4) OSTMnTOE - OSTM output enable register

This register specifies OSTMnTTOUT output mode.

Access This register can be read/written in 8-bit units.

Address <OSTMn_base> +C_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OSTMn TOE
R	R	R	R	R	R	R	R/W

Table 13-13 OSTMnTOE register contents

Bit position	Bit name	Function
0	OSTMnTOE	Specifies the OSTMnTTOUT output mode: 0: Software control mode: The level set to OSTMnTO.OSTMnTO is output to OSTMnTTOUT. 1: Timer output toggle mode: OSTMnTTOUT toggles when the interrupt request OSTMnTINT is generated.

(5) OSTMnTE - OSTM count enable status register

This register indicates whether the counter is enabled or disabled.

Access This register can be read in 8-bit units.

Address <OSTMn_base> + 10_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OSTMnTE
R	R	R	R	R	R	R	R

Table 13-14 OSTMnTE register contents

Bit position	Bit name	Function
0	OSTMnTE	Indicates, whether the counter is enabled or disabled: 0: Counter disabled 1: Counter enabled This bit is set if OSTMnTS.OSTMnTS is set or OSTMnTSST becomes 1. This bit is cleared if OSTMnTT.OSTMnTT is set.

Note If the counter is disabled, the counter value OSTMnCNT remains its value.

If the counter is restarted again, it

- restarts from the value specified by OSTMnCMP in interval timer mode.
- restarts with count value 0000 0000_H in free-run compare mode.

(6) OSTMnTS - OSTM count start trigger register

This register starts the counter.

Access This register can be written in 8-bit units. It is always read as 00_H.

Address <OSTMn_base> + 14_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OSTMnTS
R	R	R	R	R	R	R	W

Table 13-15 OSTMnTS register contents

Bit position	Bit name	Function
0	OSTMnTS	Starts the counter: 0: Ignored 1: Starts the counter and sets OSTMnTE.OSTMnTE = 1. • In interval timer mode, forced restart is executed when this bit is set while OSTMnTE.OSTMnTE = 1. • In free-run compare mode, setting this bit is ignored while OSTMnTE.OSTMnTE = 1.

(7) OSTMnTT - OSTM count stop trigger register

This register stops the counter.

Access This register can be written in 8-bit units. It is always read as 00_H.

Address <OSTMn_base> + 18_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OSTMnTT
R	R	R	R	R	R	R	W

Table 13-16 OSTMnTT register contents

Bit position	Bit name	Function
0	OSTMnTT	Stops the counter: 0: Ignored 1: Stops the counter and clears the bit OSTMnTE.OSTMnTE.

(8) OSTMnCTL - OSTM control register

This register specifies the counter operation mode and controls the generation of the interrupt request OSTMnTINT at counter start.

<R> This register can be read and written. However, it is write-only when OSTMnTE is 0 and it is read-only when OSTMnTE is 1.

Access This register can be read/written in 8-bit units. It can only be written when the counter is disabled (OSTMnTE.OSTMnTE = 0).

Address <OSTMn_base> + 20_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	OSTMn MD1	OSTMn MD0
R	R	R	R	R	R	R/W	R/W

Table 13-17 OSTMnCTL register contents

Bit position	Bit name	Function
1	OSTMnMD1	Specifies the counter operation mode: 0: Interval timer mode 1: Free-run compare mode
0	OSTMnMD0	Controls the OSTMnTINT interrupt request at counter start: 0: Disables interrupt at counter start 1: Enables interrupt at counter start

Chapter 14 Window Watchdog Timer A (WDTA)

This chapter describes window watchdog timer A (WDTA).

The first section describes all properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names.

The subsequent sections describe the features that apply to all implementations.

14.1 V850E2/Sx4-H WDTA Features

Instances

This microcontroller has the following number of instances of WDTA:

Table 14-1 Instances of WDTA

WDTA	
Number of instances	2
Names	WDTA0, WDTA1

Instances index n

Throughout this chapter, the individual instances of WTDA are identified by the index “n” (n = 0 to 1), for example, WDTAnWDTE for the WDTAn watchdog timer enable register.

Register addresses

All WDTAn register addresses are given as addresses offset from the individual base address <WDTAn_base> of each WDTAn.

The <WDTAn_base> addresses of each WDTAn are listed in the following table:

Table 14-2 Register base addresses <WDTAn_base>

WDTAn	<WDTAn_base> address
WDTA0	FF80 8000_H
WDTA1	FF80 9000_H

Clock supply

The following clocks are supplied to WDTA:

Table 14-3 WDTAn clock supply

WDTAn	Clock	Connected to:
WDTA0	WDTATCKI	Clock generator CKSCLK_A07
	PCLK	Clock generator CKSCLK_A02
WDTA1	WDTATCKI	Clock generator CKSCLK_007
	PCLK	Clock generator CKSCLK_005

Interrupts and reset outputs

WDTAn can generate the following interrupts and reset outputs:

Table 14-4 WDTA interrupts and reset outputs

WDTAn signals	Function	Connected to:
WDTA0RES	WDTA0 error reset	Reset controller WDTA0RES
WDTA0TNMI	WDTA0 error NMI	Interrupt controller WDTA0TNMI
INTWDTA0	WDTA0 75% interrupt	Interrupt controller INTWDTA0
WDTA1RES	WDTA1 error reset	Reset controller WDTA1RES
WDTA1TNMI	WDTA1 error NMI	Interrupt controller WDTA1TNMI
INTWDTA1	WDTA1 75% interrupt	Interrupt controller INTWDTA1

WDTATRTYP

The connections of the WDTA reset type input signals are listed in the table below.

Table 14-5 WDTARTYPE connections

WDTAn signals	Connected to:
WDTA0RTYPE	Reset controller WDTATRTYP
WDTA1RTYPE	Reset controller WDTATRTYP

WDTA hardware reset

WDTA and its registers are initialized by the following reset signal:

Table 14-6 WDTAn reset signal

WDTAn	Reset signal
WDTA0	System reset SYSRES

14.2 WDTA Startup Options

The startup options determine the startup configuration of WDTA after reset release. They are described in the following table.

Table 14-7 WDTA startup options

Startup option	Function	Description	Connected to:
OPWDEN	WDTA enabler/ disabler	Enables/disables WDTA: 0: WDTA is disabled 1: WDTA is enabled	• WDTA0 Flash option OPBT0.OPBT0[19] • WDTA1 Flash option OPBT0.OPBT0[23]
OPWDOVF[2:0]	Count clock setting	Specifies the reset value of the count clock control bits WDTAnMD.WDTAnOVF[2:0].	• WDTA0/WDTA1 Flash option OPBT0.OPBT0[18:16]
OPWDTPR	Start mode signal selector	Specifies the signal that sets the start mode: 0: OPWDRUN startup option 1: WDTATRTYP input signal If WDTATRTYP is selected (OPWDTPR = 1), the start mode depends on the reset type. See *14.4.1 "WDTA after reset release"* for details.	Fixed to 0
OPWDRUN	Automatic start enabler	Specifies the start mode: 0: Software trigger start mode 1: Automatic start mode OPWDRUN only applies if OPWDTPR = 0. See *14.4.1 "WDTA after reset release"* for details.	• WDTA0 Flash option OPBT0.OPBT0[20] • WDTA1 Flash option OPBT0.OPBT0[24]
OPWDVAC	Variable Activation Code (VAC) enabler	Enables/disables the Variable Activation Code function (VAC) 0: VAC is disabled 1: VAC is enabled See *14.4.2 "WDTA trigger"* for details.	• WDTA0 Flash option OPBT0.OPBT0[22] • WDTA1 Flash option OPBT0.OPBT0[26]
OPWDWS[1:0]	Initial open window size setting	Specifies the reset value of the open window size control bits WDTAnMD.WDTAnWS[1:0]. The open window size control bits only apply after the first WDTA trigger and not after reset release. After reset release the open window size is 100%. See *14.4.5 "Window function"* for details.	Fixed to 11_B
<R> OPWDINT	INTWDTAn (75% interrupt) request generation	Specifies the reset value of control bit WDTAnMD.WDTAnWIE. This bit enables/ disables the output of the 75% interrupt request INTWDTAn. See *14.4.4 "75% interrupt output"* for details.	Fixed to 0

Note The configuration of the SCIT board for both WDTAs will be provided later.

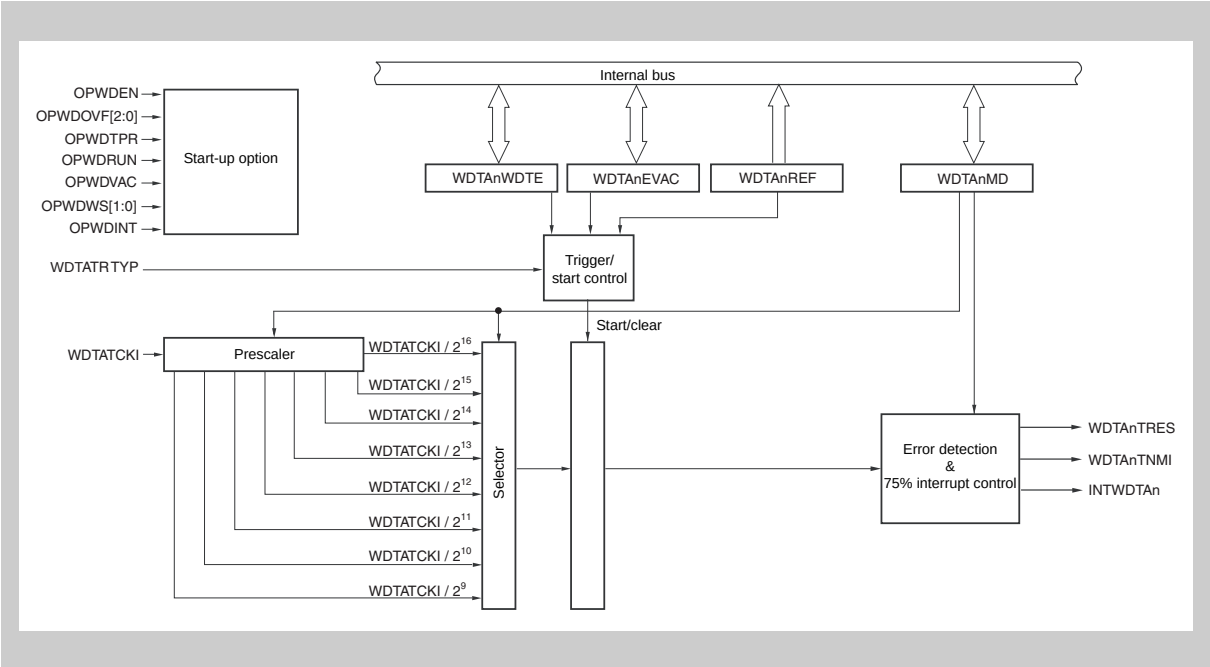
14.3 Functional Overview

Features summary

WDTA has the following functions:

- Operation mode after reset selectable by using startup option
- Fixed activation code and variable activation code (VAC) selectable
- Two start modes available:
  - Automatic start mode
  - Software trigger mode
- Operation upon error detection selectable:
  - Generation of NMI request WDTAnTNMI on error detection
  - Generation of reset WDTAnTRES on error detection
- Interrupt request generation at 75% of the counter overflow value
- Window function

The following figure shows the main components of WDTA:



<R> **Figure 14-1 Block diagram of WDTA**

14.4 Functional Description

WDTA generates a reset or a non-maskable interrupt if the 16-bit counter overflows or if any other error condition is fulfilled. For a description of all error conditions, see *14.4.3 “Error detection” on page 587*.

The counter is cleared and restarted every time a WDTA trigger occurs in the open window period. See *14.4.2 “WDTA trigger” on page 586* and *14.4.5 “Window function” on page 590* for details.

At 75% of the maximum counter value, WDTA can generate an interrupt request INTWDTAn. See *14.4.4 “75% interrupt output” on page 589* for details.

<R> After reset release, the startup options specify the start mode and the WDTA settings. The settings can be modified by writing the watchdog timer mode register WDTAnMD. See *14.4.1 “WDTA after reset release” on page 583* for details.

14.4.1 WDTA after reset release

(1) Start modes

WDTA provides two modes for the counter start after reset release:

- Software trigger start mode

  The counter value remains 0000_H after reset release.

  The counter is started with the first WDTA trigger.

- Automatic start mode

  The counter starts automatically after reset release.

(2) Start mode selection

The start mode can be selected as follows:

- By startup options
- By the WDTATRTYP input signal

  This signal indicates the reset type. Thus, the selected start mode after reset release depends on the reset type.

The start mode selection is listed in the following table.

Table 14-8 Start mode selection

Startup options		Input signal	Reset type	Start mode
OPWDTPR	OPWDRUN	WDTATRTYP		
0	0	Ignored	Ignored	Software trigger
	1			Automatic
1	0	Ignored	Ignored	Software trigger
	1	0	Any apart from automatic start reset source	Software trigger
		1	Automatic start reset source	Automatic

(3) WDTA settings after reset release

The WDTA settings are as follows between reset release and the first trigger:

Function	Setting	Remark
Start mode	Specified by startup options	For a description of the start modes, see *14.4.1 "WDTA after reset release" on page 583*.
Count clock		
75% interrupt mode		
Error mode	Reset mode	Any error condition before the first trigger generates a reset.
Open window size	100%	If automatic start mode is specified, the first trigger is valid any time before the counter overflows.

Change WDTA settings

After the first trigger, WDTA continues according to the settings of the Watchdog Timer mode register WDTAnMD.

To change the WDTA settings, WDTAnMD must be written *before* the first trigger. Changing the value of WDTAnMD *after* the first trigger leads to an error.

If WDTAnMD is not changed before the first trigger, the WDTA mode is specified by the initial value of WDTAnMD.

The new or initial value of WDTAnMD applies after the first trigger.

Automatic start mode timing

The automatic start mode timing and the changes to the WDTA settings are illustrated in the following figure.

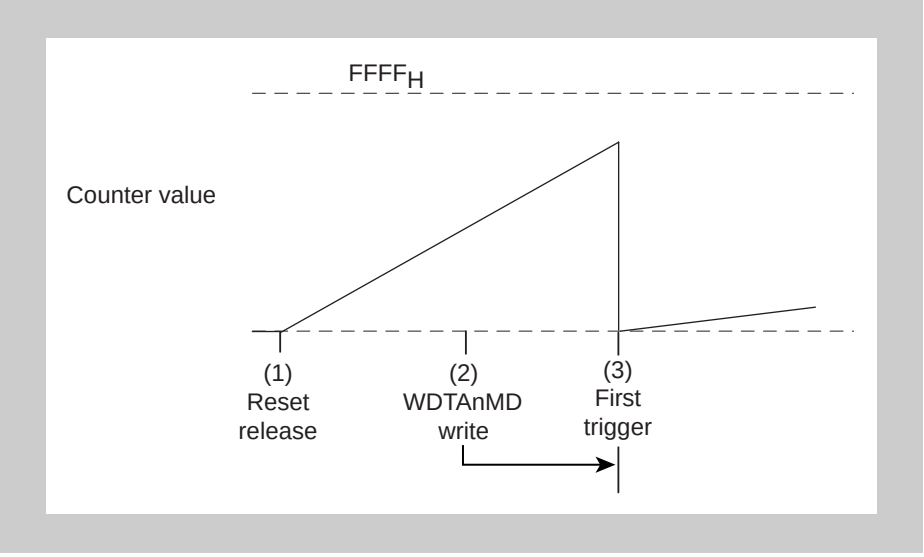


Figure 14-2 Timing diagram of WDTA start in automatic start mode

The timing diagram above shows the following:

1. After reset release, the counter starts immediately.
   The count clock is specified by the startup options, for example:
   – Count clock after reset release = WDTATCKI / 2^{13}
   (OPWDOVF[2:0] = 100_B)

2. WDTAnMD is written before the first trigger. However, the settings are not applied immediately.
3. The first trigger must occur before the counter overflows.

   After the first trigger, the settings specified in WDTAnMD are applied, for example a new count clock.

<R> – Count clock after first trigger = WDTATCKI / 2^{16} (WDTAnMD.WDTAnOVF[2:0] = 111_B

Software trigger start mode timing

The software trigger start mode timing and the changes to the WDTA settings are illustrated in the following figure.

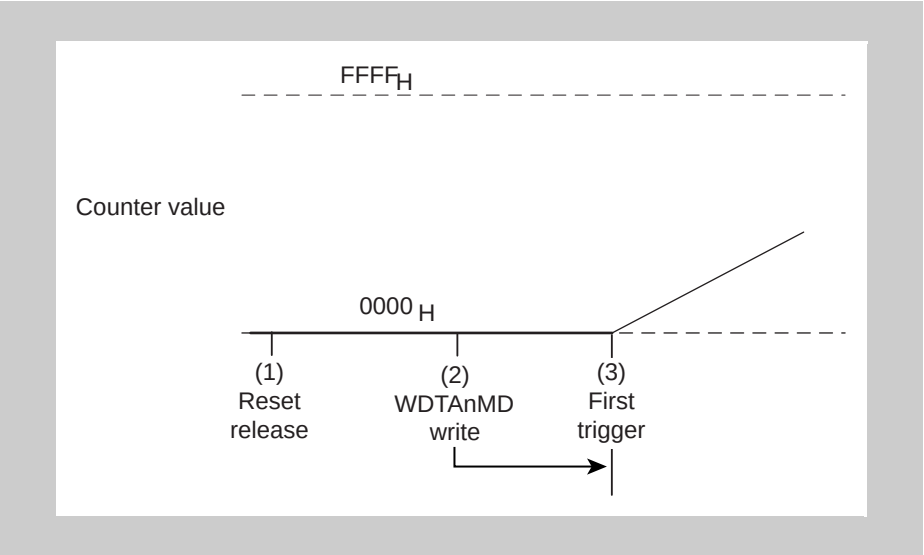


Figure 14-3 Timing diagram of WDTA start in software trigger start mode

The timing diagram above shows the following:

1. After reset release, the counter remains 0000_H until the first trigger.
   The count clock is specified by the startup options, but it does not have any effect.
2. WDTAnMD is written before the first trigger. However, the settings are not applied immediately.
3. The counter starts at the first trigger.

   The count clock and other settings specified in WDTAnMD are applied.

14.4.2 WDTA trigger

The WDTA trigger has the following functions:

- Counter start trigger in software trigger start mode
- Counter restart trigger to avoid counter overflow

The trigger register used differs depending on whether the activation code is fixed or variable. The type of activation code and the associated trigger register are specified by using the startup option OPWDVAC.

Table 14-9 Trigger register and activation code

Type of activation code	Trigger register	Activation code
Fixed	WDTAnWDTE	AC_H
Variable	WDTAnEVAC	See *(1) "Variable activation code calculation" on page 586* for details.

(1) Variable activation code calculation

The variable activation code (ExpectWDTE) is calculated using a reference value in register WDTAnREF. The reference value in WDTAnREF is updated each time the trigger register WDTAnEVAC is written.

Use the expression below to calculate the variable activation code (ExpectWDTE):

ExpectWDTE = AC_H – WDTAnREF (old)

Use the expression below to calculate how the WDTAnREF value is updated:

WDTAnREF (new) = rotate left 1 bit (ExpectWDTE)

The table below lists the variable activation codes according to the number of triggers.

Table 14-10 Variable activation code development

No[a]	WDTAnREF (old)		ExpectWDTE (AC_H - WDTAnREF)		WDTAnREF (new)	
0	0000 0000	00_H	1010 1100	AC_H	0101 1001	59_H
1	0101 1001	59_H	0101 0011	53_H	1010 0110	$A6_H$
2	1010 0110	$A6_H$	0000 0110	06_H	0000 1100	$0C_H$
…	…	…	…	…	…	…

a) Number of triggers after reset

Note Bit 7 of the WDTAnEVAC register (WDTAnEVAC7) cannot be cleared to 0 after WDTA has been started. Thus even if bit 7 of the activation code is 0, WDTA will not stop.

14.4.3 Error detection

The conditions for error detection are:

- Overflow interval time is exceeded (counter overflow)
- Wrong activation code is written to the trigger register
- Writing to the trigger register outside the open window.
- Illegal update of Watchdog Timer mode register WDTAnMD:
  - Writing a *new* value to WDTAnMD after the first trigger leads to an error detection.
  - Writing the same value to WDTAnMD after the first trigger does *not* lead to an error detection.

Error mode When an error is detected, either an NMI request (WDTAnTNMI) or a reset (WDTAnTRES) is generated.

WDTAnMD.WDTAnERM selects the error mode:

- WDTAnMD.WDTAnERM = 0: NMI mode
- WDTAnMD.WDTAnERM = 1: reset mode

<R> **Note** If an error is detected before the first WDTA trigger, a WDTAnTRES is generated, according to the default configuration.

The following figure shows the reset or NMI request generation when the counter overflows and automatic start mode is selected.

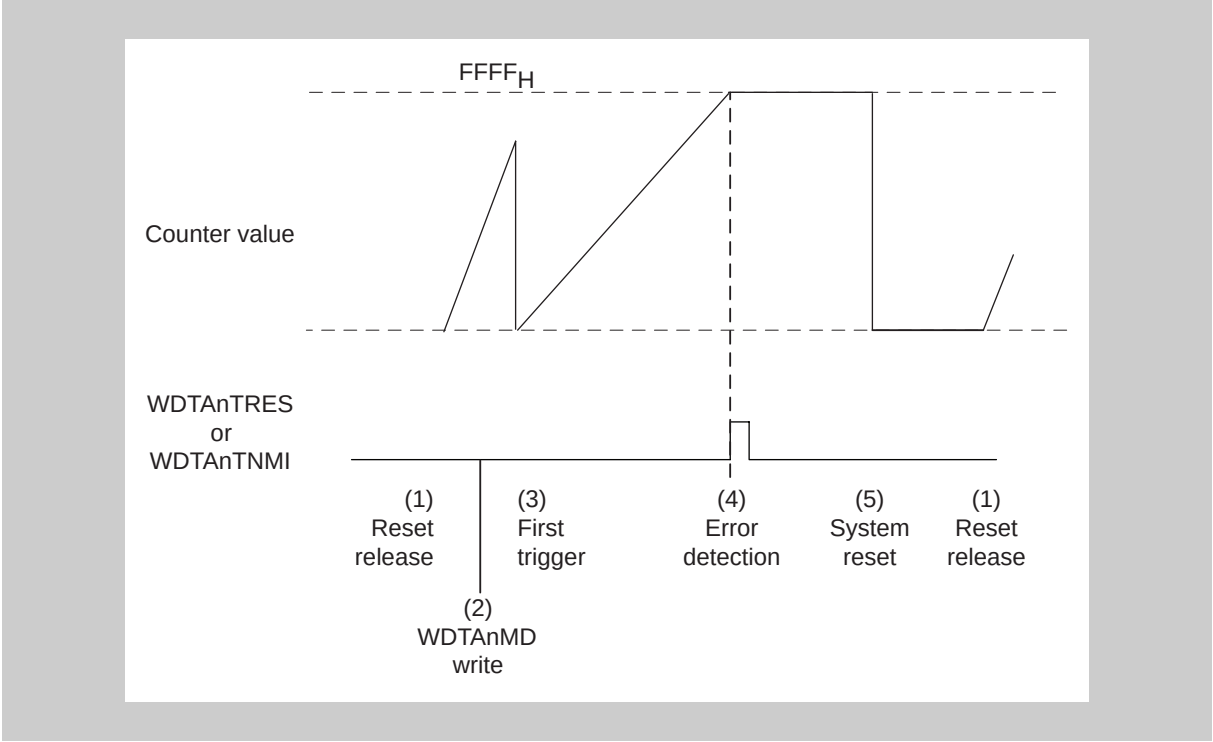


Figure 14-4 Timing diagram of WDTA NMI request or reset generation

The timing diagram above shows the following:

1. After reset release, the counter starts (automatic start mode is selected).
2. WDTAnMD is written before the first trigger. However, the settings are not applied immediately.
3. The counter is cleared at the first trigger and the new WDTA settings are applied.
4. When the counter overflows, an error is detected. Depending on the error mode, either interrupt request WDTAnTNMI or reset WDTAnTRES is generated.

   The counter value remains until the system reset is performed.
5. When the system is reset, the counter is cleared and stopped until reset release.

14.4.4 75% interrupt output

When the counter reaches 75% of the maximum counter value, the interrupt request INTWDTAn is generated.

This function can be automatically enabled with the startup option OPWDINT = 1.
This function can be enabled or disabled by specifying a setting in the WDTAnMD.WDTAnWIE register.

The following figure shows 75% interrupt request generation under the following conditions:

- Automatic start mode selected
- Count clock changes after first trigger

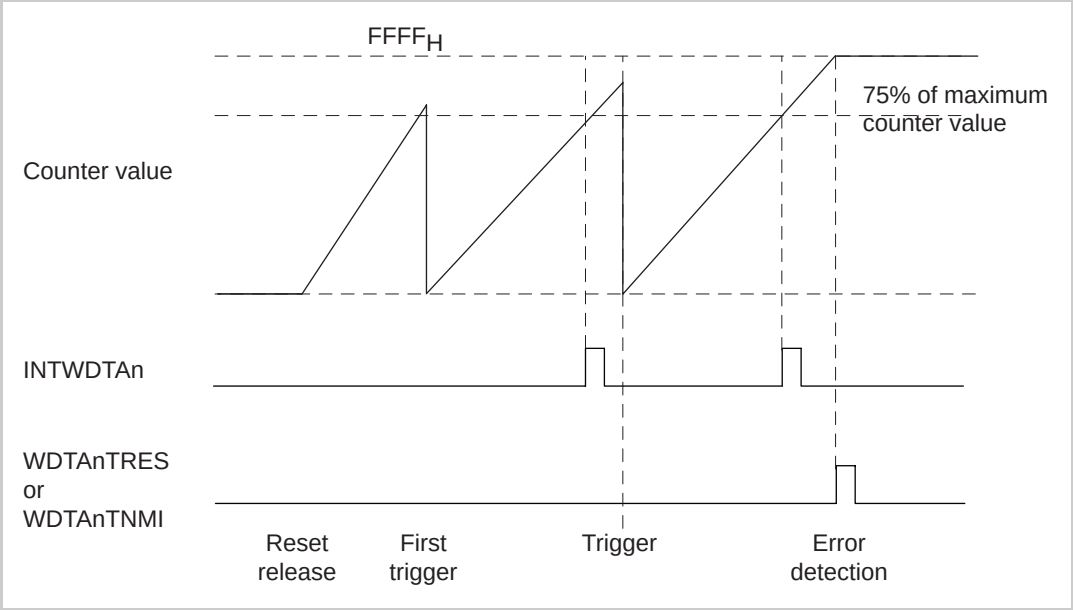


Figure 14-5 Timing diagram of WDTA 75% interrupt output

14.4.5 Window function

When the open window size is set to less than 100%, an error is detected if the trigger occurs outside the open window.

The definition of the open window size differs before and after the first trigger:

- After reset release, the open window size is 100%.

  OPWDWS[1:0] and bits WDTAnMD.WDTAnWS[1:0] are not applied.
- After the first trigger, the open window size is specified by bits WDTAnMD.WDTAnWS[1:0].

The following figure shows WDTA operation with an open window size of 25% and with automatic start mode selected.

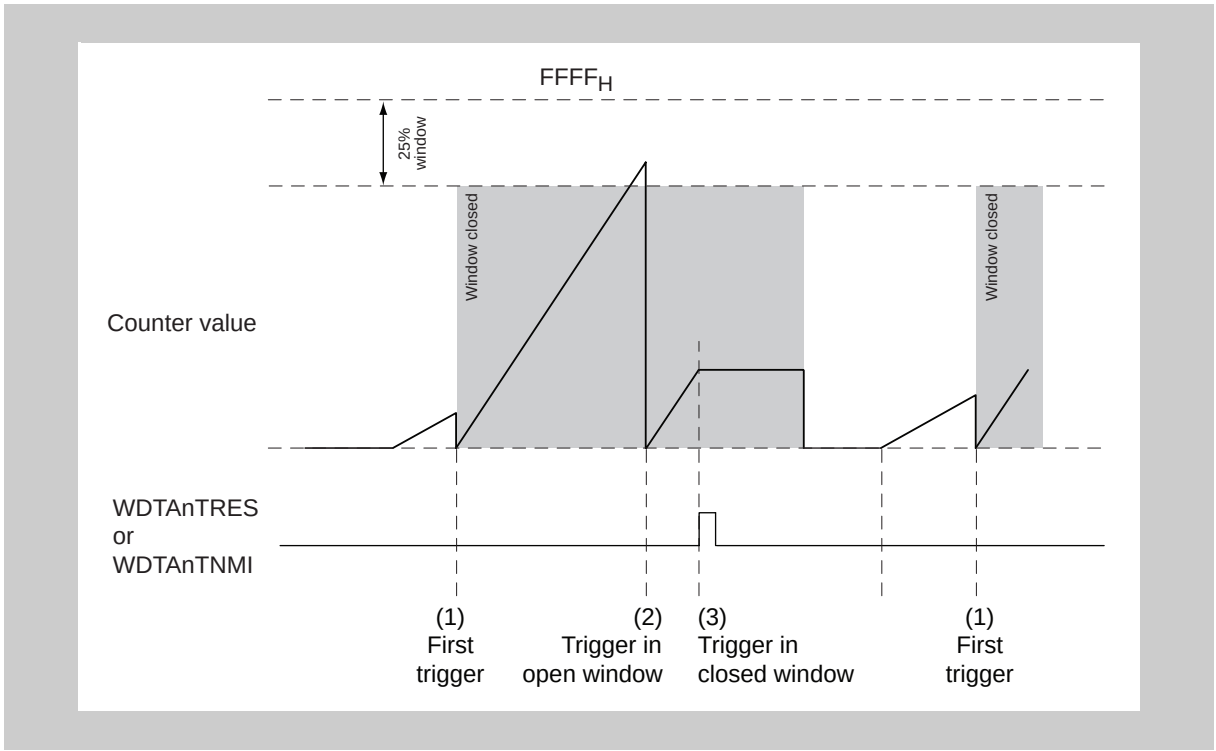


Figure 14-6 Timing diagram of WDTA window function

The timing diagram above shows the following:

1. The open window size is fixed to 100% for the first trigger.
2. A trigger that occurs in the open window does not generate an error.
3. A trigger that occurs in the closed window generates a WDTAnTNMI request or a WDTAnTRES reset, depending on the selected error mode.

14.5 Registers

This section contains a description of all registers of WDTA.

14.5.1 WDTA register overview

WDTA is controlled and operated by the following registers:

Table 14-11 WDTA register overview

Register name	Symbol	Address
WDTA enable register	WDTAnWDTE	<WDTAn_base> + 0000_H
WDTA enable register for VAC	WDTAnEVAC	<WDTAn_base> + 0004_H
WDTA reference value register	WDTAnREF	<WDTAn_base> + 0008_H
WDTA mode register	WDTAnMD	<WDTAn_base> + $000C_H$

14.5.2 WDTA register details

(1) WDTAnWDTE – WDTA enable register

This register is the WDTA start control and trigger register if the VAC function is not used (startup option OPWDVAC = 0).

WDTA trigger Writing AC_H to this register restarts the counter. See *14.4.2 "WDTA trigger" on page 586* for details.

The behaviour of this register depends on activation of the VAC function, see *Table 14-14 "WDTAnWDTE behaviour"*.

Access This register can be read or written in 8-bit units.

Address <WDTAn_base> + 0000_H

Initial Value Depends on startup options OPWDEN, OPWDTPR, WDTATRTYP, OPWDRUN and OPWDVAC. See *Table 14-13 "WDTAnRUN initial value"*.

<R> This register is initialized by any reset.

7	6	5	4	3	2	1	0
WDTAnRUN	0	1	0	1	1	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 14-12 WDTAnWDTE register contents

Bit position	Bit name	Function
7	WDTAnRUN	Enables/disables WDTAn: 0: WDTAn disabled 1: WDTAn enabled Because WDTA cannot be stopped once it was started, this bit can only be cleared by a reset.

Initial value This bit is only valid if WDTA is enabled (OPWDEN = 1) and VAC is disabled (OPWDVAC = 0). In this case, the initial value of bit WDTAnRUN depending on other startup options is listed below:

<R> **Table 14-13 WDTAnRUN initial value**

Startup options				Input signal	Initial value of WDTAnRUN
OPWDEN	OPWDVAC	OPWDPR	OPWDRUN	WDTATRTYP	
0	Ignored	Ignored	Ignored	Ignored	0
1	1	Ignored	Ignored	Ignored	0
	0	0	0	Ignored	0
			1	Ignored	1
		1	0	Ignored	0
			1	0	0
				1	1

The behaviour of WDTAnWDTE during read/write accesses depends on activation of the VAC mode, as shown in the table below.

Table 14-14 WDTAnWDTE behaviour

OPWDVAC	WDTAnWDTE		Remark
	Read	Write	
0	AC_H	WDTA trigger AC_H[a]	VAC disabled: WDTAnWDTE enabled
1	$2C_H$	Ignored	VAC enabled: WDTAnWDTE disabled

a) Any other write value will lead to an error detection.

(2) WDTAnEVAC – WDTA enable VAC register

This register is the start control and trigger register if the VAC function is used (startup option OPWDVAC = 1).

WDTA trigger Writing the correct activation code to this register restarts the counter. See *14.4.2 "WDTA trigger" on page 586*.

The behaviour of this register depends on activation of the VAC function, see *Table 14-17 "WDTAnEVAC behaviour"*.

Access This register can be read or written in 8-bit units.

Address <WDTAn_base> + 0004_H

Initial Value Depends on startup options OPWDEN, OPWDTPR, WDTATRTYP, OPWDRUN and OPWDVAC. See *Table 14-15 "WDTAnEVAC register contents"*.

<R> This register is initialized by any reset.

7	6	5	4	3	2	1	0
WDTAnEVAC7	0	1	0	1	1	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 14-15 WDTAnEVAC register contents

Bit position	Bit name	Function
7	WDTAnEVAC7	Enables or disables WDTAn. 0: Disable WDTAn. 1: Enable WDTAn. Because WDTA cannot be stopped once it was started, this bit can only be cleared by a reset. Thus even if bit 7 of the activation code is 0, WDTA will not stop.

Initial value This bit is only valid if WDTA is enabled (OPWDEN = 1) and VAC is enabled (OPWDVAC = 1). In this case, the initial value of bit WDTAnEVAC7 depending on other startup options is listed below:

<R> **Table 14-16 WDTAnEVAC7 initial value**

Startup options		Input signal	Start mode	Initial value of WDTAnEVAC7
OPWDTPR	OPWDRUN	WDTATRTYP		
0	0	Ignored	Software trigger	0
0	1	Ignored	Default	1
1	Ignored	0	Software trigger	0
1	Ignored	1	Default	1

The behaviour of WDTAnEVAC during read/write accesses depends on activation of the VAC mode, as shown in the table below.

Table 14-17 WDTAnEVAC behaviour

OPWDVAC	WDTAnEVAC		Remark
	Read	Write	
0	$2C_H$	ignored	VAC disabled: WDTAnEVAC disabled
1	last written VAC	WDTA trigger VAC[a]	VAC enabled: WDTAnEVAC enabled

a) Any other write value will lead to an error detection.

(3) WDTAnREF – WDTA reference value register

This register contains the reference value for calculating the activation code of the VAC function. It is automatically updated after every trigger operation. See *14.4.2 "WDTA trigger" on page 586*.

If VAC is disabled (OPWDVAC = 0), reading this register returns 00_H.

Access This register can be read in 8-bit units.

Address <WDTAn_base> + 0008_H

<R> **Initial Value** 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
WDTAnREF[7:0]							
R	R	R	R	R	R	R	R

Table 14-18 WDTAnREF register contents

Bit position	Bit name	Function
7 to 0	WDTAnREF[7:0]	Reference value for activation code calculation.

(4) WDTAnMD – WDTA mode register

This register is used to specify the overflow interval time, the 75% interrupt output mode, the error mode, and the open window size.

It can be updated only once after reset release and before the first trigger. The updated value is effective from the next WDTA trigger.

Updating this register after WDTA has been started leads to error detection, but the read value of this register can be written without generating an error.

Access This register can be read or written in 8-bit units.

Address <WDTAn_base> + $000C_H$

Initial Value Depends on startup options OPWDOVF[2:0], OPWDINT and OPWDWS[1:0]. See *14.2 "WDTA Startup Options"*.

<R> This register is initialized by any reset.

	7	6	5	4	3	2	1	0
	0	WDTAnOVF[2:0]			WDTAnWIE	WDTAnERM	WDTAnWS[1:0]	
<R>	R/W[a]	R/W	R/W	R/W	R/W	R/W	R/W	R/W

a) Writing to this bit is ignored, reading returns 0.

Table 14-19 WDTAnMD register contents (1/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>6 to 4</td><td>WDTAnOVF[2:0]</td><td>Selects the overflow interval time:
<table>
<tr><th>WDTAnOVF2</th><th>WDTAnOVF1</th><th>WDTAnOVF0</th><th>Overflow interval time</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>WDTATCKI / 2^9</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>WDTATCKI / 2^{10}</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>WDTATCKI / 2^{11}</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>WDTATCKI / 2^{12}</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>WDTATCKI / 2^{13}</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>WDTATCKI / 2^{14}</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>WDTATCKI / 2^{15}</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>WDTATCKI / 2^{16}</td></tr>
</table>
The reset values of WDTAnOVF[2:0] depend on startup option OPWDOVF[2:0].</td></tr>
<tr><td>3</td><td>WDTAnWIE</td><td>Enables/disables the 75% interrupt request INTWDTAn:
0: INTWDTAn disabled
1: INTWDTAn enabled
The reset value of WDTAnWIE depends on startup option OPWDINT.</td></tr>
</table>

Table 14-19 WDTAnMD register contents (2/2)

<R>

Bit position	Bit name	Function
2	WDTAnERM	Specifies the error mode: 0: NMI request mode 1: Reset mode (default)
1, 0	WDTAnWS[1:0]	Selects the open window size: WDTAnWS1 / WDTAnWS0 / Open window size 0 / 0 / 25% 0 / 1 / 50% 1 / 0 / 75% 1 / 1 / 100% The reset values of WDTAnWS[1:0] depend on startup option OPWDWS[1:0].

Chapter 15 Timer Array Unit A (TAUA)

This chapter describes timer array unit A (TAUA).

The first section describes all V850E2/Sx4-H specific properties, such as instances, register base addresses, and input/output signal names.
The subsequent sections describe the features that apply to all implementations.

15.1 V850E2/Sx4-H TAUA Features

Instances

This microcontroller has following number of instances of TAUA:

Table 15-1 Instances of TAUA

TAUA	
Number of instances	5
Name	TAUA0 to TAUA4

Instances index n

Throughout this chapter, the individual instances of TAUA is identified by the index "n" (n = 0), for example, TAUAnTOM for the TAUAn channel output mode register.

Channel index m

TAUA has 16 channels. Throughout this chapter, the individual channels are identified by the index “m” (m = 0 to 15), thus a certain channel is denoted as CHm.
The even numbered channels (m = 0, 2, 4, 6, 8, 10, 12, 14) are denoted as CHm_even.
The odd numbered channels (m = 1, 3, 5, 7, 9, 11, 13, 15) are denoted as CHm_odd.

Register addresses

All TAUAn register addresses are given as addresses offset from the individual base address <TAUAn_base>.
The base address <TAUAn_base> of each TAUAn is listed in the following table:

Table 15-2 Register base addresses <TAUAn_base>

TAUAn	<TAUAn_base> address
TAUA0	FF80 8000_H

Clock supply

The following clock is input to TAUA:

Table 15-3 TAUAn clock supply

TAUAn	Clock	Connected to:
TAUA0	PCLK	Clock generator CKSCLK_006

Interrupts and DMA

TAUA can generate the following interrupt and DMA requests:

Table 15-4 TAUAn interrupt and DMA requests

TAUAn signals	Function	Connected to:
TAUA0:		
INTTAUA0I0	Channel 0 interrupt	Interrupt controller INTTAUA0I0 DMA controller trigger 70
INTTAUA0I1	Channel 1 interrupt	Interrupt controller INTTAUA0I1 DMA controller trigger 71
INTTAUA0I2	Channel 2 interrupt	Interrupt controller INTTAUA0I2 DMA controller trigger 72
INTTAUA0I3	Channel 3 interrupt	Interrupt controller INTTAUA0I3 DMA controller trigger 73
INTTAUA0I4	Channel 4 interrupt	Interrupt controller INTTAUA0I4 DMA controller trigger 76
INTTAUA0I5	Channel 5 interrupt	Interrupt controller INTTAUA0I5 DMA controller trigger 79
INTTAUA0I6	Channel 6 interrupt	Interrupt controller INTTAUA0I6 DMA controller trigger 80
INTTAUA0I7	Channel 7 interrupt	Interrupt controller INTTAUA0I7 DMA controller trigger 84
INTTAUA0I8	Channel 8 interrupt	Interrupt controller INTTAUA0I8 DMA controller trigger 12
INTTAUA0I9	Channel 9 interrupt	Interrupt controller INTTAUA0I9 DMA controller trigger 13
INTTAUA0I10	Channel 10 interrupt	Interrupt controller INTTAUA0I10 DMA controller trigger 14
INTTAUA0I11	Channel 11 interrupt	Interrupt controller INTTAUA0I11 DMA controller trigger 15
INTTAUA0I12	Channel 12 interrupt	Interrupt controller INTTAUA0I12 DMA controller trigger 16
INTTAUA0I13	Channel 13 interrupt	Interrupt controller INTTAUA0I13 DMA controller trigger 17
INTTAUA0I14	Channel 14 interrupt	Interrupt controller INTTAUA0I14 DMA controller trigger 18
INTTAUA0I15	Channel 15 interrupt	Interrupt controller INTTAUA0I15 DMA controller trigger 19

TAUA hardware reset

TAUA and its registers are initialized by the following reset signal:

Table 15-5 TAUAn reset signal

TAUAn	Reset signal
TAUA0	System reset SYSRES

I/O signals The I/O signals of TAUA are listed in the following table:

Table 15-6 TAUAn I/O signals

TAUA signal	Function	Connected to:
TAUA0TTIN0	Channel 0 input	Port TAUA0I0/FCN0 TSOUT[a]/ port URTE10RX[a]
TAUA0TTIN1	Channel 1 input	Port TAUA0I1/FCN1 TSOUT[a]
TAUA0TTIN2	Channel 2 input	Port TAUA0I2/port IISA0SCKI[a]
TAUA0TTIN3	Channel 3 input	Port TAUA0I3/port IISA1SCKI[a]
TAUA0TTIN4 to TAUA0TTIN15	Channel 4 to 15 input	Ports TAUA0I4 to TAUA0I415
TAUA0TTOUT0 to TAUA0TTOUT15	Channel 0 to 15output	Ports TAUA0O0 to TAUA0O15

a) For details,see *15.2 “TAUA Input Selection”*.

All TAUAn interrupt and I/O signals are sketched in the following figure.

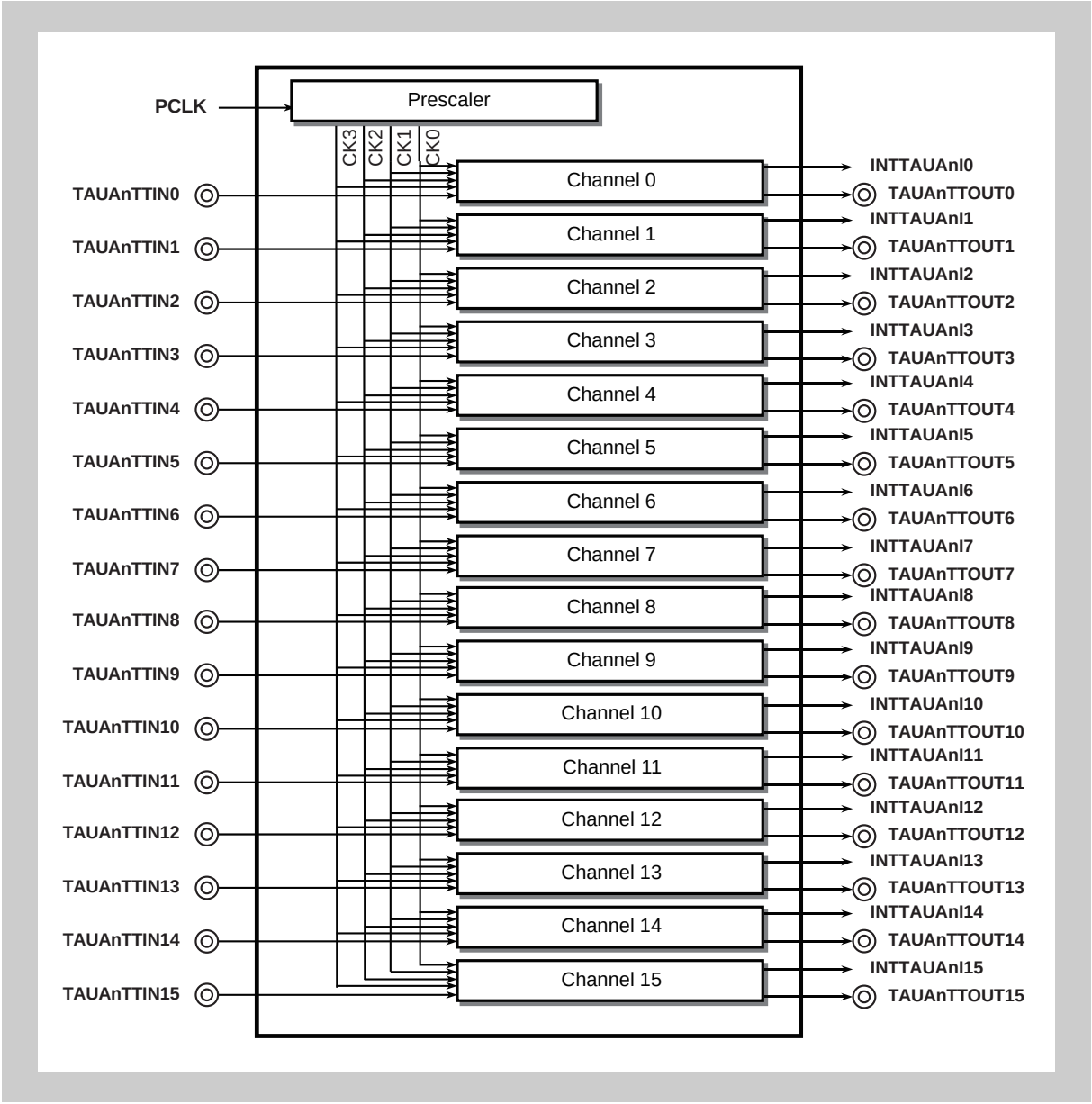


Figure 15-1 TAUA I/O and interrupt signals

15.2 TAUA Input Selection

15.2.1 TAUA0 input selection

The TAUA0 has several options to connect its input signals:

- FCN0 and FCN1 time stamp output signals TSOUT for timing measurements
- URTE10 and URTE11 data receive signals URTE10RX and URTE11RX for baud rate measurement

The following figure depicts the TAUA0 input selection scheme:

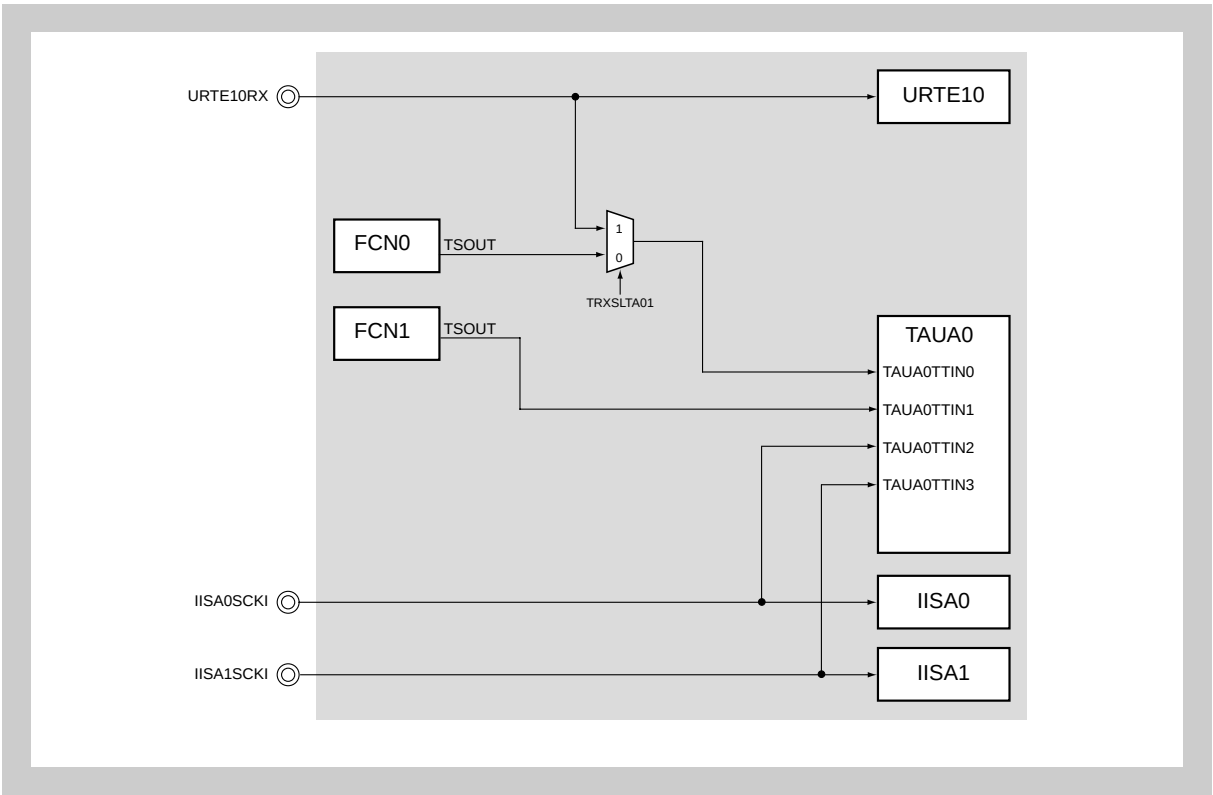


Figure 15-2 TAUA0 input selector

The following tables show the optional inputs to several TAUAn inputs:

Table 15-7 TAUA0 input selections - TAUA0TTIN0 to TAUA0TTIN3

TAUA0 input	Input options	Selection control
		TRXSLTA0 register bits
TAUA0TTIN0	FCN0 TSOUT (CAN I/F 0 time stamp output)	TRXSLTA0[1:0] = 00_B
	Port URTE10RX (UART10 data receive signal)	TRXSLTA0[1:0] = 11_B
TAUA0TTIN1	FCN1 TSOUT (CAN I/F 1 time stamp output)	TRXSLTA02 = 1_B

Table 15-7 TAUA0 input selections - TAUA0TTIN0 to TAUA0TTIN3 (continued)

TAUA0 input	Input options	Selection control
		TRXSLTA0 register bits
TAUA0TTIN2	IISA0SCKI (IISA0 serial clock intput)	TRXSLTA06 = 1_B
TAUA0TTIN3	IISA1SCKI (IISA1 serial clock intput)	TRXSLTA07 = 1_B

(1) TRXSLTA0 - TAUA0 receive input selection register

This register selects the input signals to several TAUAn inputs out of signals related to other functional modules (FCN, URTE, and IISA).

Access This register can be read/written in 8-bit units.

Address FF77 1004_H

Initial Value 00_H

7	6	5	4	3	2	1	0
TRXSLTA07	TRXSLTA06	0	0			TRXSLTA0[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-8 TRXSLTA0 register contents

Bit position	Bit name	Function
7	TRXSLTA07	Selection of TAUA0TTIN3 1: IISA1SCKI
6	TRXSLTA076	Selection of TAUA0TTIN2 1: IISA0SCKI
2	TRXSLTA072	Selection of TAUA0TTIN1 1: FCN1 TSOUT
1, 0	TRXSLTA0[1:0]	Selection of TAUA0TTIN0

TRXSLTA0[1:0]	TAUA0TTIN0
01_B	FCN0 TSOUT
11_B	Port URTE10RX

15.3 Functional Overview

Features summary

The TAUA has the following functions:

- 16 channels
- 16-bit counter and 16-bit data register per channel
- Independent channel operation
- Synchronous channel operation (master and slave operation)
- Generation of different types of output signal
- Real-time output
- Counter can be triggered by external signal
- Interrupt generation

The following figure shows the main components of the TAUA:

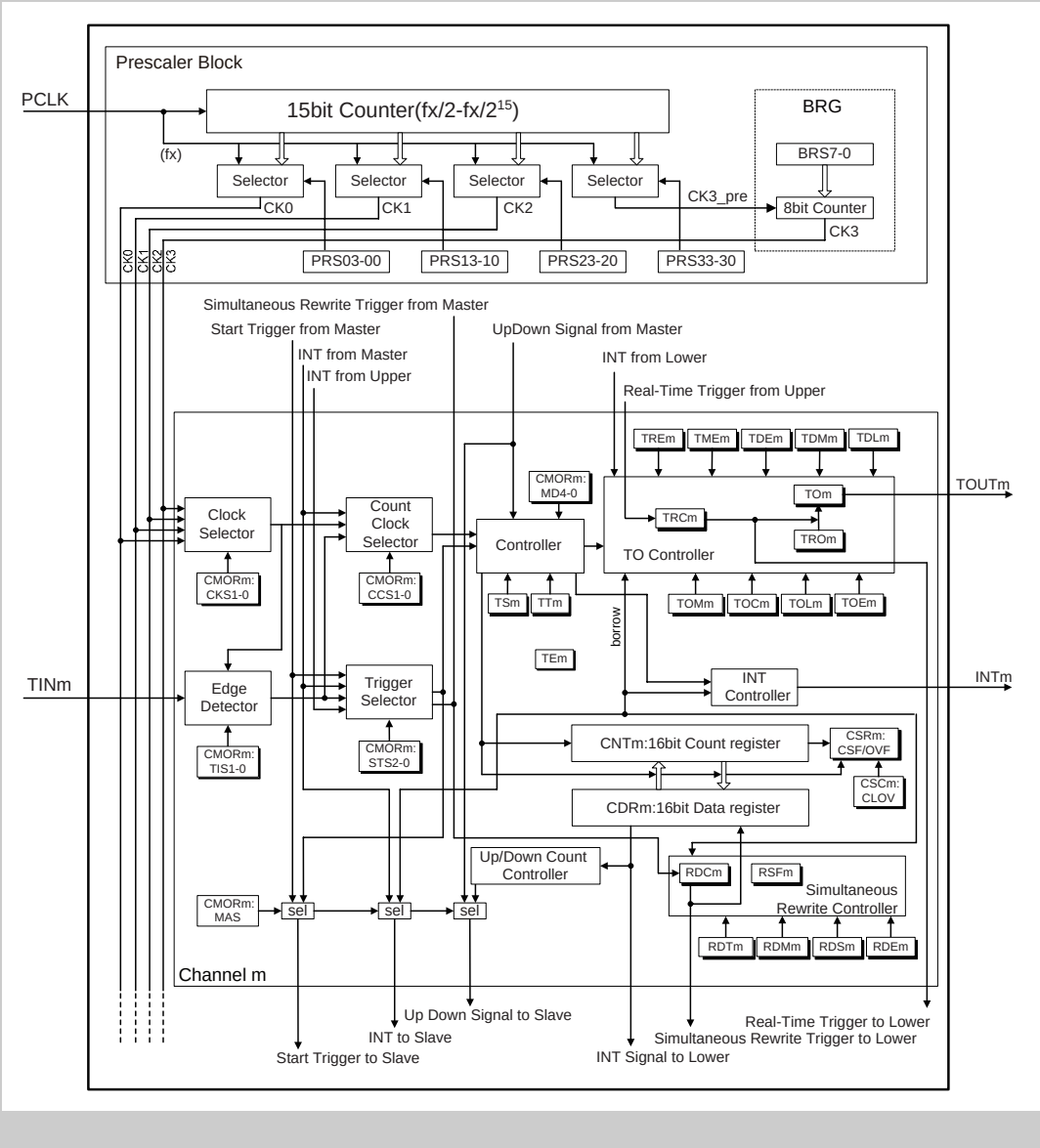


Figure 15-3 Block diagram of the TAUA

The prefix “TAUAn” has been omitted from the register names for the sake of clarity in the above figure.

15.3.1 Terms

In this chapter, the following terms are used:

- **Independent / synchronous channel operation**

  Independent or synchronous channel operation describes the dependency of channels on each other:

  – If a channel operates independent of all other channels, this is called independent channel operation.

  – If a channel operates depending on other channels, this is called synchronous channel operation.

- **Channel group**

  In synchronous channel operation, all channels that depend on each other are referred to as a “channel group”.

  A channel group has one master channel and one or more slave channels.

- **Operation mode**

  An operation mode can be selected for every channel m. The operation mode defines the *basic* operation and features of a channel.

  In synchronous channel operation, every channel in the channel group can operate in a different operation mode.

  Examples are capture mode, event count mode, and interval timer mode.

- **Channel output mode**

  The channel output mode defines the operation of TAUAnTTOUTm

  – of a single channel (independent output operation) or

  – of all channels in a channel group (synchronous output operation).

  Examples are independent channel output mode 1 and synchronous channel output mode 2 with dead time output.

- **Channel operation function**

  The channel operation function defines the *complete* function and all features

  – of a single channel (independent channel operation) or

  – of all channels in a channel group (synchronous channel operation).

- **Upper / lower channel**

  Depending on the channel number m, a neighboring channel can be referred to as “upper” or “lower” channel:

  – Upper channel: Channel with a smaller channel number

  – Lower channel: Channel with a higher channel number

  Example:
  For channel 5, channel 3 is an upper channel and channel 9 is a lower channel.

15.4 Functional Description

TAUA is used to perform various count or timer operations and to output a signal which depends on the result of the operation. It contains one prescaler for count clock generation and 16 channels, each equipped with a 16 bit counter TAUAnCNTm and a 16-bit data register TAUAnCDRm to hold the start or compare value of the counter.

It also contains several control and status registers.

Independent and synchronous operation

Every channel can operate in different operation modes, either independently or in combination with other channels (synchronously). If a channel group has one master channel and one or more slave channels, the slave channels depend on the master channel.

When a channel is operated independently, its operation mode and functions are not affected by those of other channels. When a channel is operated synchronously it is either a master or a slave. A master channel can have multiple slaves, and the state of one channel affects that of the other channels. For example, this means that one channel can control when another starts to count, is reset, etc.

The following describes the functional blocks:

Prescaler

The prescaler provides up to 4 clock signals (CK0 to CK3) that can be used as count clocks for all channels.

For count clocks CK0 to CK2, a clock obtained by dividing PCLK by 2^0 to 2^{15} using the prescaler can be selected. For the fourth count clock CK3, a division factor that is not a power of 2 can be specified by using BRG.

Clock and count clock selection

For every channel, the count clock selector selects which of the following is used as the clock source:

- One of the clocks CK0 to CK3 (selected by the clock selector)
- INTTAUAnIm from master channel
- TAUAnTTINm input signal valid edge

Controller

The controller controls the main operations of the counter:

- Operation mode (selected by bits TAUAnCMORm.TAUAnMD[4:0])
- Counter start enable (TAUAnTS.TAUAnTSm) and counter stop (TAUAnTT.TAUAnTTm)

  When counter start is enabled, status flag TAUAnTE.TAUAnTEm is set.
- Count direction (up/down) (can be controlled by master channel)

Trigger selector

Depending on the selected operation mode, the counter starts automatically when it is enabled (TAUAnTE.TAUAnTEm = 1), or it waits for an external start trigger signal. Any of the following signals can be used as the start trigger:

- Synchronous channel start trigger input TAUAnTSSTm
- TAUAnTTINm input valid edge
- INTTAUAnIm from the master or any upper channel
- Up/down output trigger signal of the master channel
- Dead-time output signal of the TAUAnTTOUTm generation unit.

Simultaneous rewrite controller Simultaneous rewrite control is a function that can be used in synchronous operation modes. The data registers (TAUAnCDRm) of all channels in a channel group can be rewritten at any time. The simultaneous rewrite controller ensures that new data register values of all channels become effective at the same time.

TAUAnTO controller The output control of every channel enables the generation of various output signal forms such as PWM signals or triangular waves.

15.4.1 Timer operation functions

The functions below can be achieved by operating TAUA independently on each channel or by operating it on a combination of multiple channels.

Table 15-9 TAUA operation functions (1/2)

Independent channel operation function	Synchronous channel operation function
Independent channel operation functions	Synchronous channel operation functions
Interval timer function	PWM output function
TAUAnTTINm input interval timer function	Trigger start PWM output function
Delay count function	Delay pulse output function
One-pulse output function	A/D conversion trigger output function type 1
Independent channel signal measurement functions	Synchronous PWM signal functions triggered by an external signal
TAUAnTTINm input pulse interval measurement function	One-shot pulse output function
TAUAnTTINm input signal width measurement function	Offset trigger output function
Overflow interrupt output function (during TAUAnTTINm width measurement)	Synchronous triangle PWM output function
TAUAnTTINm input period count detection function	Triangle PWM output function
Overflow interrupt output function (during TAUAnTTINm input period count detection)	Triangle PWM output function with dead time
TAUAnTTINm input pulse interval judgment function	A/D conversion trigger output function type 2
TAUAnTTINm input signal width judgment function	Synchronous non-complementary modulation output function and synchronous complementary modulation output function
Independent channel real-time functions	Non-complementary modulation output function type 1
Real-time output function type 1	Non-complementary modulation output function type 2
Real-time output function type 2	Complementary modulation output function
Independent channel simultaneous rewrite functions	Other independent channel functions
Simultaneous rewrite trigger generation function type 1	Interrupt culling function
Simultaneous rewrite trigger generation function type 2	–
Independent channel one-phase PWM function	–

Table 15-9 TAUA operation functions (2/2)

Independent channel operation function	Synchronous channel operation function
One-phase PWM output function	–
Other independent channel functions	–
External event count function	–
Clock divide function	–
TAUAnTTINm input position detection function	–

15.5 General Operating Procedure

The following lists the general operation procedure for the TAUAn:

After reset release, the operation of each channel is stopped. Clock supply is started and writing to each register is enabled. The control register of TAUAnTTOUTm is also initialized and outputs a low level.

1. Set the TAUAnTPS and TAUAnBRS registers to specify the clock frequency of CK0 to CK3.
2. Configure the desired TAUAn function:
   – Set the operation mode
   – Set the channel output mode
   – Set any other control bits
3. Enable the counter by setting the TAUAnTS.TAUAnTSm bit to 1.

   The counter starts to count immediately, or when an appropriate trigger is detected, depending on the bit settings.
4. During counting, if desired, and if possible for the configured function, stop the counter or perform a forced restart operation.
5. Stop the function by setting the TAUAnTT.TAUAnTTm bit to 1.

Note A detailed description of the required control bits and the operation of the individual functions is given in *15.15 “Independent Channel Operation Functions” on page 647* and *15.22 “Synchronous Channel Operation Functions” on page 769*.

15.6 Operation Modes

The TAUA contains 12 operation modes.

One operation mode can be set for each channel. It is specified using the TAUAnCMOR.TAUAnMD[4:0] bits.

Note For registers and bits, some values are fixed according to the operation function, and others are selected by the user.

For details about the settings for registers and bits, see the chapters describing each operation function.

15.7 Concepts of Synchronous Channel Operation

In synchronous channel operation, multiple channels depend on each other, or are affected by changes in another channel. Therefore, several rules apply for the use of synchronous channel functions. These rules are detailed in *15.7.1 "Rules"*.

Two special features for synchronous channel operation are detailed in the following subchapters:

- *15.7.2 "Simultaneous start and stop of synchronous channel counters" on page 614*
- *15.8 "Simultaneous Rewrite" on page 615*

15.7.1 Rules

Number of masters and slaves

- Only even channels (CH0, CH2, CH4, …) can be set as master channels. Any channel except CH0 can be set as a slave channel.
- Only channels lower than the master channel can be set as slave channels, and several slave channels can be set for one master channel.
  Example: If CH2 is a master channel, CH3 and the lower channels (CH4, CH5, …) can be set as slave channels.
- If multiple master channels are used, slave channels cannot cross the master channels.
  Example: If CH0 and CH4 are master channels, CH1 to CH3 can be set as slave channels for CH0, but CH5 to CH15 cannot.

Operation clock

- The same operation clock must be set for the corresponding slave channel and the master channel. Specify the same value for the TAUAnCMORm.TAUAnCKS[1:0] bits of the synchronized master and slave channels.

The basic concepts of master/slave usage and operation clocks are illustrated in the following figure.

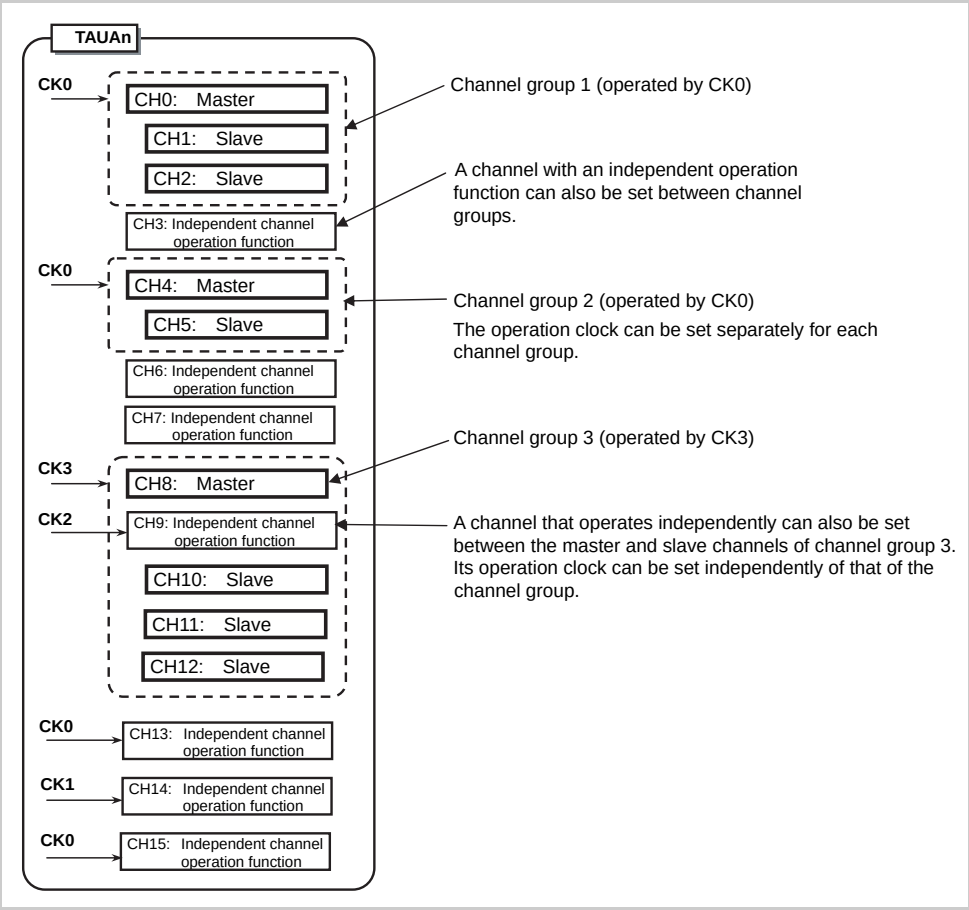


Figure 15-4 Grouping of the channels and assignment of operation clocks

Control trigger signals of the master and slave channels

- A master channel can output a control trigger signal to its slave channels.
- Slave channels can use the control trigger signal of the master channel but cannot transfer their control trigger signals to the lower channels.
- A master channel cannot use the control trigger signal of an upper master channel.

15.7.2 Simultaneous start and stop of synchronous channel counters

Channels that are operated synchronously can be started and stopped simultaneously, both within a unit, and between units.

(1) Simultaneous start and stop within a unit

- To simultaneously start synchronized channels, the TAUAnTS.TAUAnTSm bits of the channels must be set at the same time.
- To simultaneously stop synchronized channels, the TAUAnTT.TAUAnTTm bits of the channels must be set at the same time.

Setting a TAUAnTS.TAUAnTSm bit to 1 sets the corresponding TAUAnTE.TAUAnTEm bit to 1, enabling counting. When the counter starts counting depends on the operation mode.

(2) Simultaneous start between units

Counters in different units can also be started simultaneously if the corresponding counters are enabled before receiving the simultaneous trigger signal.

15.8 Simultaneous Rewrite

15.8.1 Introduction

Simultaneous rewrite describes the ability to change the compare/start value and the output logic of multiple channels at the same time.

The corresponding data and control registers (TAUAnCDRm and TAUAnTOLm) can nevertheless be written at any time. The new value does not affect the counter operation or the output signal until simultaneous rewrite is triggered.

Simultaneous rewrite can be triggered by:

- The counter on the master channel or upper channel (depending on the selected operation mode) reaching a certain value
- INTTAUAnIm being issued on the upper channel specified by TAUAnRDC.TAUAnRDCm

There are four methods for simultaneous rewrite. These are listed in the following table, along with how to specify them and when they cause simultaneous rewrite to be triggered.

Table 15-10 Simultaneous rewrite methods and when they are triggered

Method	Simultaneous rewrite triggered when	TAUAn RDE. TAUAn RDEm	TAUAn RDS. TAUAn RDSm	TAUAn RDM. TAUAn RDMm
-	No simultaneous rewrite	0	0	0
A	The master channel (re)starts counting.	1	0	0
B	The slave channel starts counting down at the upper peak of a triangular cycle.	1	0	1
C1	INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.TAUAnRDCm.	1	1	0
C2	INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.TAUAnRDCm that in turn is triggered by an external signal.	1	1	0

The following table lists which of these four methods is available for each channel operation function. For details about the individual channel operation functions, see the corresponding sections in *15.15 “Independent Channel Operation Functions” on page 647* and *15.22 “Synchronous Channel Operation Functions” on page 769*.

Table 15-11 Channel operation functions and methods they use

Function	–	A	B	C1	C2
Simultaneous rewrite trigger output function type 1				X	
PWM output function		X		X	
One-shot pulse output function		X			
Trigger start PWM output function		X			X
Offset trigger output function	X				
Delay pulseoutput function		X			
Triangle PWM output function			X	X	
Triangle PWM output function with dead time			X	X	
Interrupt culling function		X	X	X	
AD conversion trigger output function type 1		X		X	
AD conversion trigger output function type 2			X	X	
Non-complementary modulation output function type 1		X		X	
Non-complementary modulation output function type 2			X	X	
Complementary modulation output function			X	X	

15.8.2 How to control simultaneous rewrite

The following figure shows the general procedure for simultaneous rewrite. The three main blocks (Initial setup, start counter & count operation, and simultaneous rewrite) are explained afterwards.

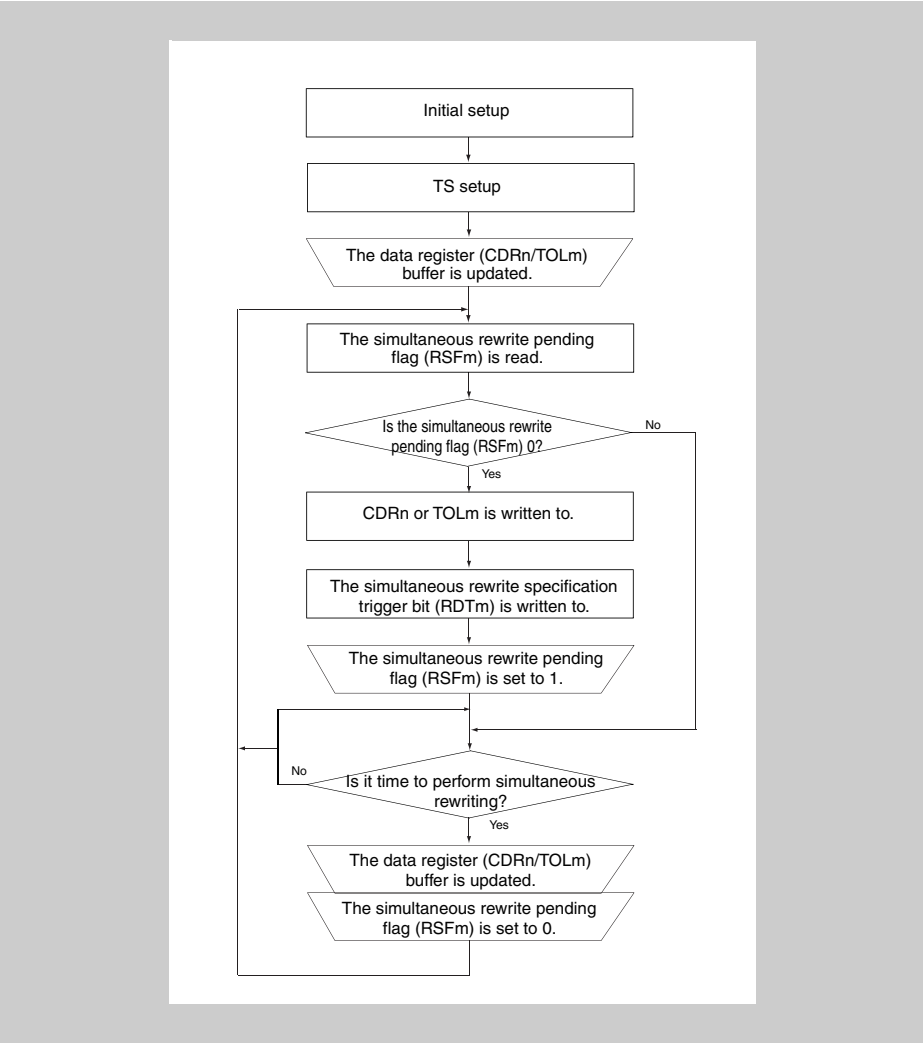


Figure 15-5 General procedure for simultaneous rewrite

(1) Initial settings

- To enable simultaneous rewrite in channel m, set TAUAnRDE.TAUAnRDEm = 1
- To select the type of simultaneous rewrite, set TAUAnRDM.TAUAnRDMm and TAUAnRDS.TAUAnRDSm according to the values in *Table 15-10 "Simultaneous rewrite methods and when they are triggered" on page 615*.
- Specify the simultaneous rewrite trigger generation channel for TAUAnRDC.TAUAnRDCm. (Prerequisite: TAUAnRDS.TAUAnRDSm is specified as an upper channel.)

(2) Start counter and count operation

- To start all the TAUAnCNTm counters in the channel group, set the corresponding TAUAnTS.TAUAnTSm bits to 1. TAUAnTOL.TAUAnTOLm and the values in the data registers (TAUAnCDRm) are loaded to the corresponding TAUAnTOL.TAUAnTOLm buffer (TAUAnTOL.TAUAnTOLm buf) and data buffer registers (TAUAnCDRm buf) and the counters start.
- Setting the reload data trigger bit (TAUAnRDT.TAUAnRDTm) to 1 sets the reload flag (TAUAnRSF.RSFm) to 1, enabling simultaneous rewrite. TAUAnRSF.RSFm remains 1 until simultaneous rewrite has taken place.
- When the specified trigger for simultaneous rewrite is detected, the TAUAnRSF.TAUAnRSFm bit is checked to see if simultaneous rewrite is enabled (TAUAnRSF.TAUAnRSFm = 1). If it is, simultaneous rewrite is carried out. If such writing is disabled, simultaneous rewriting is not performed, and the system awaits the detection of the next simultaneous rewrite trigger.

(3) Simultaneous rewrite

- When the simultaneous rewrite trigger is detected and simultaneous rewrite is enabled (TAUAnRSF.TAUAnRSFm = 1), the current values of the data registers are copied to their buffers. These values are then loaded to the corresponding counters and the values are applied the next time the counter starts or restarts.
- When simultaneous rewriting finishes, the TAUAnRSF.TAUAnRSFm bit is cleared to 0, and the system awaits the next simultaneous rewrite trigger.

15.8.3 Other general rules of simultaneous rewrite

The following rules also apply:

- TAUAnRDE.TAUAnRDEm, TAUAnRDS.TAUAnRDSm, TAUAnRDM.TAUAnRDMm, and TAUAnRDC.TAUAnRDCm cannot be changed while the counter is in operation (TAUAnTE.TAUAnTEm = 1).
- TAUAnTOL.TAUAnTOLm can only be rewritten during operation when in PWM output function or triangle PWM output function. For all other output functions, TAUAnTOL.TAUAnTOLm must be written before the counter starts. If it is rewritten in another function, TAUAnTTOUTm outputs an invalid wave.
- When an upper channel is used as the channel issuing the simultaneous rewrite trigger (TAUAnRDS.TAUAnRDSm = 1), the TAUAnRDC.TAUAnRDCm bit controls all the lower channels. This means that if the TAUAnRDC.TAUAnRDCm bits of CH2 and CH7 are set to 1 and the TAUAnRDC.TAUAnRDCm bits of other channels are set to 0, CH2 and CH7 serve as simultaneous rewrite trigger generation channels. CH2 controls the lower channels CH3 to CH6, and CH7 controls the lower channels CH8 to CH15.
- If simultaneous rewrite is enabled and an upper channel is selected for the simultaneous rewrite trigger (TAUAnRDE.TAUAnRDEm and TAUAnRDS.TAUAnRDSm = 1) but no upper channel is set (TAUAnRDC.TAUAnRDC[15:0] = 0), simultaneous rewrite cannot take place.

15.8.4 Types of simultaneous rewrite

In the following section the four simultaneous rewrite methods are explained using timing diagrams.

(1) Simultaneous rewrite when the master channel (re)starts counting (method A)

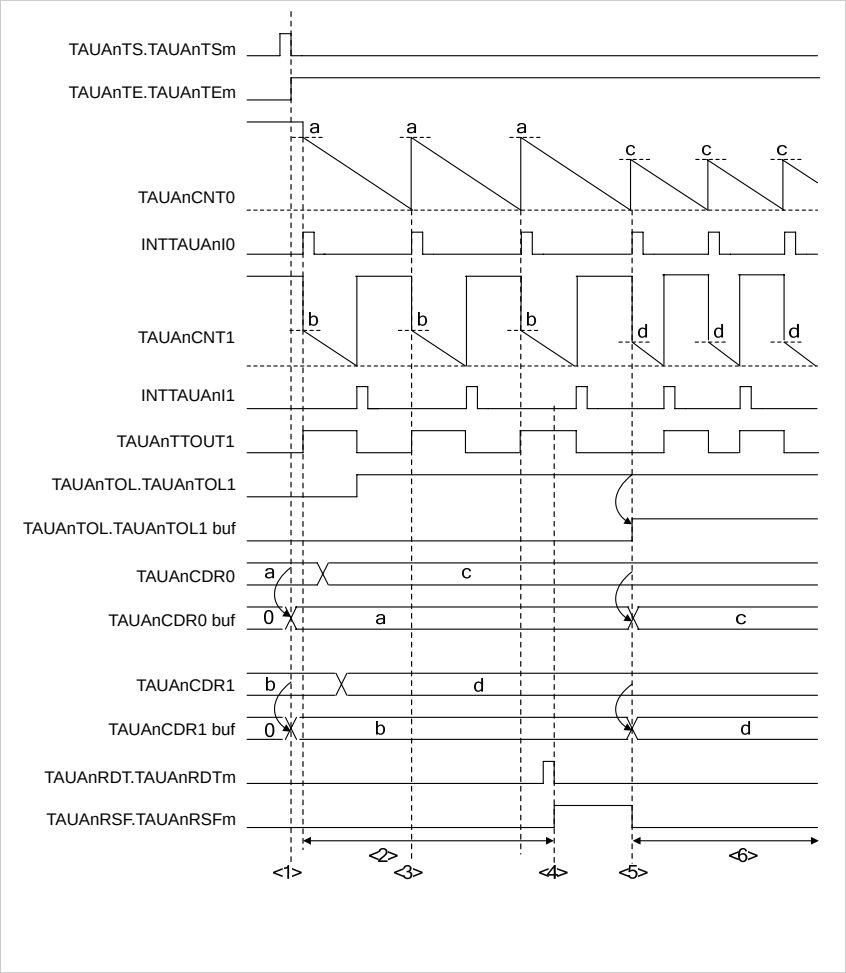


Figure 15-6 Simultaneous rewrite when the master channel (re)starts counting

Setup:

- CH0 is the master channel, counting down, CH1 represents an arbitrary slave channel, and simultaneous rewrite method A is applied.

Description:

1. When TAUAnTS.TAUAnTSm is set to 1, the value of TAUAnCDRm is copied to the TAUAnCDRm buffer, and the value of TAUAnTOL.TAUAnTOLm is copied to the TAUAnTOL.TAUAnTOLm buffer.
2. The TAUAnCDRm and TAUAnTOL.TAUAnTOLm registers can be written at any time.
3. CH0 restarts counting, but simultaneous rewrite does not occur because it is disabled (TAUAnRSF.TAUAnRSFm = 0).
4. The reload data trigger bit (TAUAnRDT.TAUAnRDTm) is set to 1 which sets the status flag (TAUAnRSF.TAUAnRSFm = 1), enabling simultaneous rewrite.
5. Because simultaneous rewriting is enabled, it is performed before counting on channel 0 resumes. The TAUAnCDRm value is loaded to the TAUAnCDRm buffer, and the TAUAnTOL.TAUAnTOLm value is loaded to the TAUAnTOL.TAUAnTOLm buffer.
6. The counters count down and await the next simultaneous rewrite trigger. The values of TAUAnCDRm and TAUAnTOL.TAUAnTOLm can be changed again.

(2) Simultaneous rewrite at the peak of a triangular cycle of the slave channel (method B)

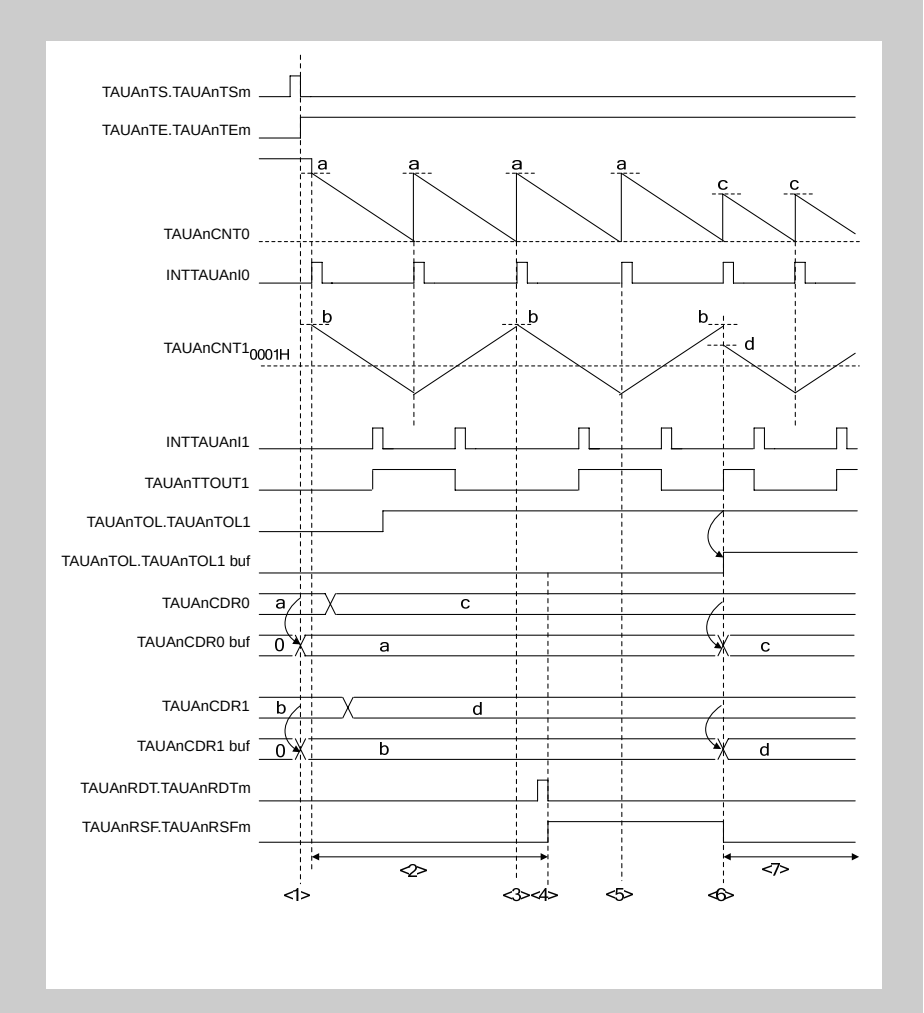


Figure 15-7 Simultaneous rewrite at the peak of a triangular cycle of the slave channel

Setup:

- CH0 is the master channel, counting up and down, CH1 represents an arbitrary slave channel, and simultaneous rewrite method B is applied.

Description:

1. When TAUAnTS.TAUAnTSm is set to 1, the value of TAUAnCDRm is copied to the TAUAnCDRm buffer.
2. The TAUAnCDRm and TAUAnTOL registers can be written at any time.
3. Simultaneous rewrite does not occur because it is disabled (TAUAnRSF.TAUAnRSFm = 0).
4. The reload data trigger bit (TAUAnRDT.TAUAnRDTm) is set to 1 which sets the status flag (TAUAnRSF.TAUAnRSFm = 1), enabling simultaneous rewrite.
5. Simultaneous rewriting does not occur at the valley of a triangular wave cycle.
6. Simultaneous rewriting is performed at the peak of a triangular wave cycle. The TAUAnCDRm value is loaded to the TAUAnCDRm buffer, and the TAUAnTOL.TAUAnTOLm value is loaded to the TAUAnTOL.TAUAnTOLm buffer.
7. The counters count down and await the next simultaneous rewrite trigger. The values of TAUAnCDRm and TAUAnTOL.TAUAnTOLm can be changed again.

(3) Simultaneous rewrite when INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.TAUAnRDCm (method C1)

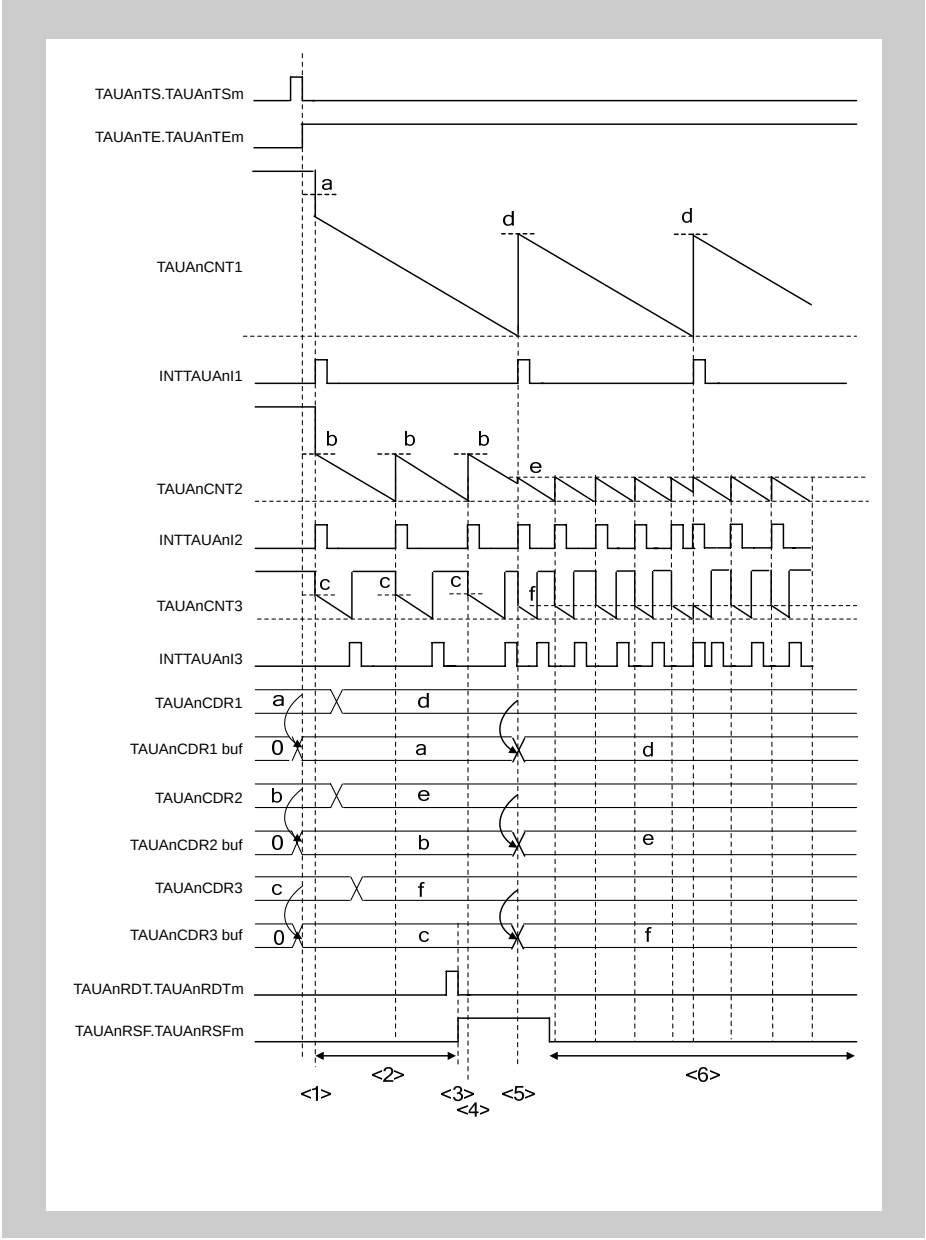


Figure 15-8 Simultaneous rewrite when INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.TAUAnRDCm

Setup:

- CH1 is an upper channel, counting down. CH2 is a master channel, CH3 is the slave channel, and simultaneous rewrite method C1 is applied. The TAUAnRDC register specifies the channel to generate the simultaneous rewrite trigger.

Description:

1. When TAUAnTS.TAUAnTSm is set to 1, the value of TAUAnCDRm is copied to the TAUAnCDRm buffer.
2. The TAUAnCDRm register can be written at any time.
3. The reload data trigger bit (TAUAnRDT.TAUAnRDTm) is set to 1 which sets the status flag (TAUAnRSF.TAUAnRSFm = 1), enabling simultaneous rewrite.
4. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by an interrupt on channel 1.
5. Simultaneous rewriting is triggered by INT1, which is caused by counter 1 reaching 0000H. The TAUAnCDRm values are loaded to the corresponding TAUAnCDRm buffers.
6. The counter counts down and awaits the next simultaneous rewrite trigger. The values of the TAUAnCDRm registers can be changed again.

(4) Simultaneous rewrite when INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.TAUAnRDCm that in turn is triggered by an external signal (method C2)

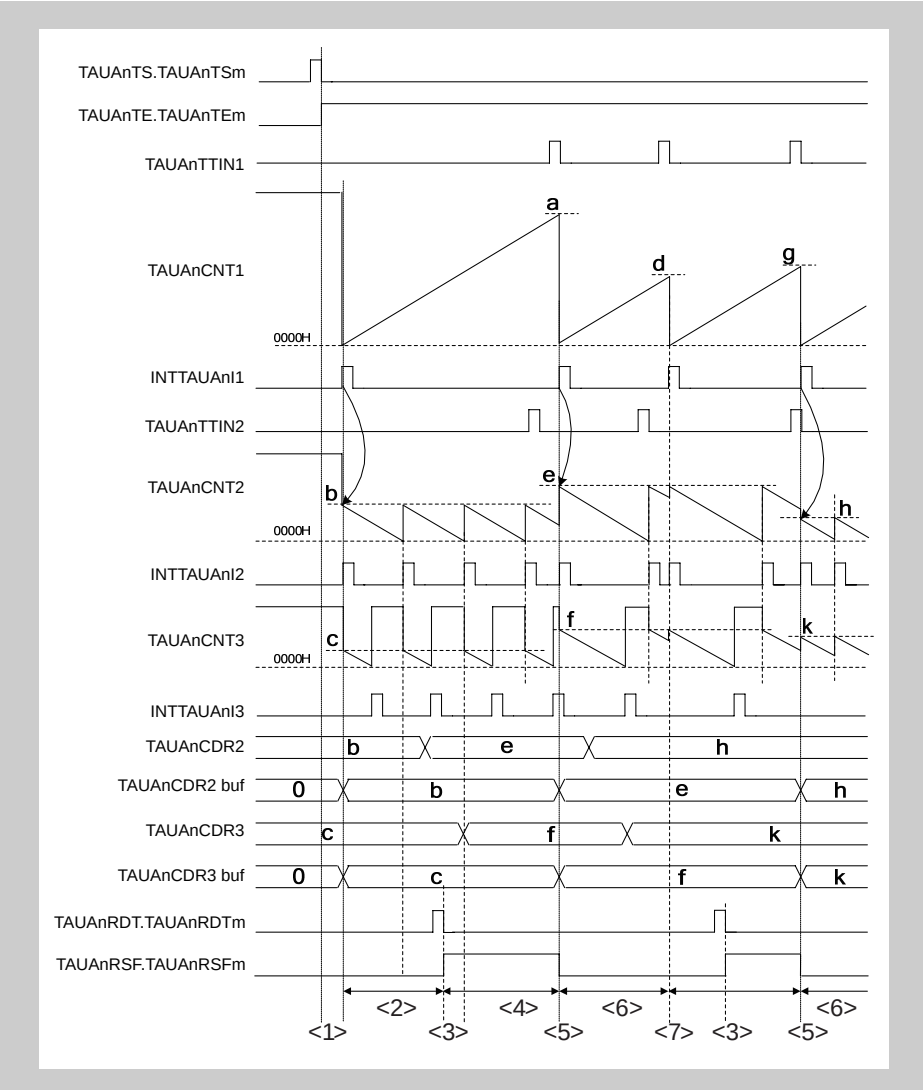


Figure 15-9 Simultaneous rewrite when INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.TAUAnRDCm that in turn is triggered by an external signal

Setup:

<R>

- CH1 is an upper channel, counting up. CH2 is a master channel, CH3 is the slave channel, and synchronous channel operation method C2 is applied. The TAUAnRDC register specifies which upper channel is monitored for an INTTAUAnIm trigger.

Description:

1. When TAUAnTS.TAUAnTSm is set to 1, the value of TAUAnCDRm is copied to the TAUAnCDRm buffer.
2. The TAUAnCDRm register can be written at any time.
3. The reload data trigger bit (TAUAnRDT.TAUAnRDTm) is set to 1 which sets the status flag (TAUAnRSF.TAUAnRSFm = 1), enabling simultaneous rewrite.
4. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by an interrupt on channel 1.
5. Simultaneous rewriting is triggered by INT1, which is caused by the external signal TIN1. The TAUAnCDRm values are loaded to the corresponding TAUAnCDRm buffers.
6. The counters count down and await the next simultaneous rewrite trigger. The values of the TAUAnCDRm registers can be changed again.
7. An external signal occurs at TIN2 but simultaneous rewrite does not take place because it is disabled (TAUAnRSF.TAUAnRSFmTAUAnRSF.TAUAnRSFm = 0).

15.9 Channel Output Modes

The output of the TAUAnTTOUTm pin can be controlled in two ways, the latter of which can be further split into individual modes:

- By software (TAUAnTOE.TAUAnTOEm = 0)

  When controlled by software, the output register bit (TAUAnTO.TAUAnTOm) can be written and the value of the bit is output from to the output pin (TAUAnTTOUTm).

- By TAUA signals (TAUAnTOE.TAUAnTOEm = 1)

  When operated by TAUA signals, the output level of TAUAnTTOUTm is set or reset or toggled by internal signals. The value of TAUAnTO.TAUAnTOm is updated accordingly to reflect the value of TAUAnTTOUTm.

  – Independently (TAUAnTOM.TAUAnTOMm = 0)

    When operated independently, the output of the TAUAnTTOUTm pin is only affected by settings of channel m. Therefore, independent channel operation must be selected (TAUAnTOM.TAUAnTOMm = 0).

  – Synchronously (TAUAnTOM.TAUAnTOMm = 1)

    When operated synchronously, the output of the TAUAnTTOUTm pin is affected by settings of channel m and those of other channels. Therefore, synchronous channel operation must be selected for all participating channels (TAUAnTOM.TAUAnTOMm = 1).

The TAUAnTO.TAUAnTOm bit can always be read to determine the current value of TAUAnTTOUTm, regardless of whether the pin is controlled by software, operated independently, or operated synchronously.

Control bits

The settings of the control bits required to select a specific channel output mode are listed in *Table 15-12 “Channel output modes” on page 628*.

The channel output modes are described in detail in

- *15.9.2 “Channel output modes controlled independently by TAUAn signals” on page 630*
- *15.9.3 “Channel output modes controlled synchronously by TAUAn signals” on page 632*.

Collective TAUAnTOm bit manipulation

Whether to apply settings to the TAUAnTOm bits is controlled using the TAUAnTOE.TAUAnTOEm bits.

When writing to the TAUAnTO register, TAUAnTOm settings are only written to channels for which the TAUAnTOE.TAUAnTOEm bit is cleared to 0. The TAUAnTOm settings are not applied for channels for which the corresponding TAUAnTOE.TAUAnTOEm bit is set to 1.

Note

The TAUAnTO.TAUAnTOm bits are allocated so that the bit numbers correspond to the channel numbers.

Output logic

Positive logic or inverted logic of the output is specified by control bit TAUAnTOL.TAUAnTOLm.

The value of the TAUAnTOL.TAUAnTOLm bit must be set before the counter is started. It can only be changed during operation in PWM output function or triangle PWM output function. Otherwise, changes to TAUAnTOL.TAUAnTOLm result in an undefined TAUAnTTOUTm signal.

See *15.8 “Simultaneous Rewrite” on page 615*.

The various channel output modes and the channel output control bits are listed in the following table.

Table 15-12 Channel output modes

Channel output mode	TAUAn TOE. TAUAn TOEm	TAUAn TOM. TAUAn TOMm	TAUAn TOC. TAUAn TOCm	TAUAn TDE. TAUAn TDEm	TAUAn TRE. TAUAn TREm	TAUAn TME. TAUAn TMEm	TAUAn TDM. TAUAn TDMm
By software							
Independent channel output mode controlled by software	0	x					
By TAUA signals, independently							
Independent channel output mode 1	1	0	0	0	0	0	0
with real-time output					1		
Independent channel output mode 2			1		0		
By TAUA signals, synchronously							
Synchronous channel output mode 1	1	1	0	0	0	0	0
with non-complementary modulation output					1	1	
Synchronous channel output mode 2			1	0	0	0	0
with dead time output				1			
with one-phase PWM output							1
with complementary modulation output					1	1	0
with non-complementary modulation output			1	0			

- All combinations not listed in this table are forbidden.
- Bits marked with an x can be set to any value.

Note The following bits cannot be changed during count operation (TAUAnTE.TAUAnTE = 1):

- TAUAnTOE.TAUAnTOEm,
- TAUAnTOM.TAUAnTOMm,
- TAUAnTOC.TAUAnTOCm,
- TAUAnTDE.TAUAnTDEm,
- TAUAnTRE.TAUAnTREm, and
- TAUAnTDM.TAUAnTDMm

The following bits cannot be changed during count operation (TAUAnTE.TAUAnTEm = 1) except in channel output modes with modulation output:

- TAUAnTME.TAUAnTMEm,
- TAUAnTDL.TAUAnTDLm

15.9.1 General procedure for specifying a channel output mode

The following steps describe the general procedure for specifying a TAUAnTTOUTm channel output mode. The prerequisite is that timer output operation is disabled (TAUAnTOE.TAUAnTOEm = 0).

1. Set TAUAnTO.TAUAnTOm to specify the initial level of the TAUAnTTOUTm output.
2. Set the channel output mode using *Table 15-12 “Channel output modes” on page 628* and the output logic using the TAUAnTOL.TAUAnTOLm bit.
3. Start the counter (TAUAnTS.TAUAnTSm = 1).

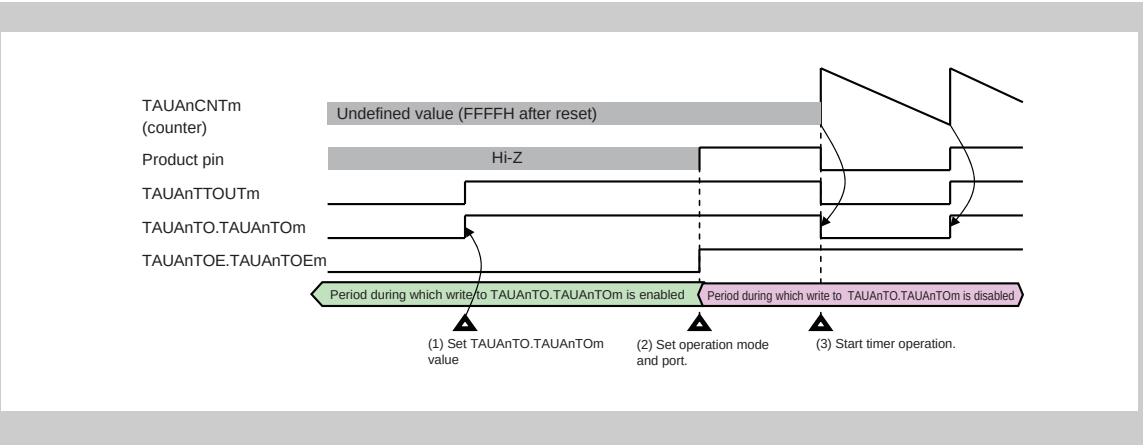


Figure 15-10 General procedure for specifying a TAUAnTTOUTm channel output mode

15.9.2 Channel output modes controlled independently by TAUAn signals

This chapter lists the channel output modes that are controlled independently by TAUAn signals. The control bits used to specify a mode are listed in *Table 15-12 “Channel output modes” on page 628*.

(1) independent channel output mode 1

Set/reset conditions In this output mode, TAUAnTTOUTm toggles when INTTAUAnIm is detected. The value of TAUAnTOL.TAUAnTOLm is ignored.

Prerequisites None, other than those in *Table 15-12 “Channel output modes” on page 628*.

(2) independent channel output mode 1 with real-time output

In this output mode, the value of the TAUAnTRO.TAUAnTROm bit of the trigger channel is output to TAUAnTTOUTm. The trigger channel is specified by setting the corresponding TAUAnTRC.TAUAnTRCm bit to 1. It controls all lower channels for which TAUAnTRC.TAUAnTRCm = 0.

Set/reset conditions The value of the TAUAnTRO.TAUAnTROm bit is only sent to TAUAnTTOUTm when an interrupt INTTAUAnIm occurs on the trigger channel. The interrupt is generated either:

- at certain specified intervals or
- on detection of a valid TAUAnTTINm input edge / counter start

The type of trigger is set using the TAUAnCMORm.TAUAnMD[4:1] bits.

Prerequisites Both master and slave channels can be set as a trigger generation channel. A channel for which TAUAnTRC.TAUAnTRCm is set to 1 serves as the trigger generation channel even if TAUAnTRE.TAUAnTREm is set to 0. Real-time output cannot take place if there is no channel for which TAUAnTRE.TAUAnTREm is set to 1 or if TAUAnTRCm.TRC0 = 0.

This can be seen in the following figure.

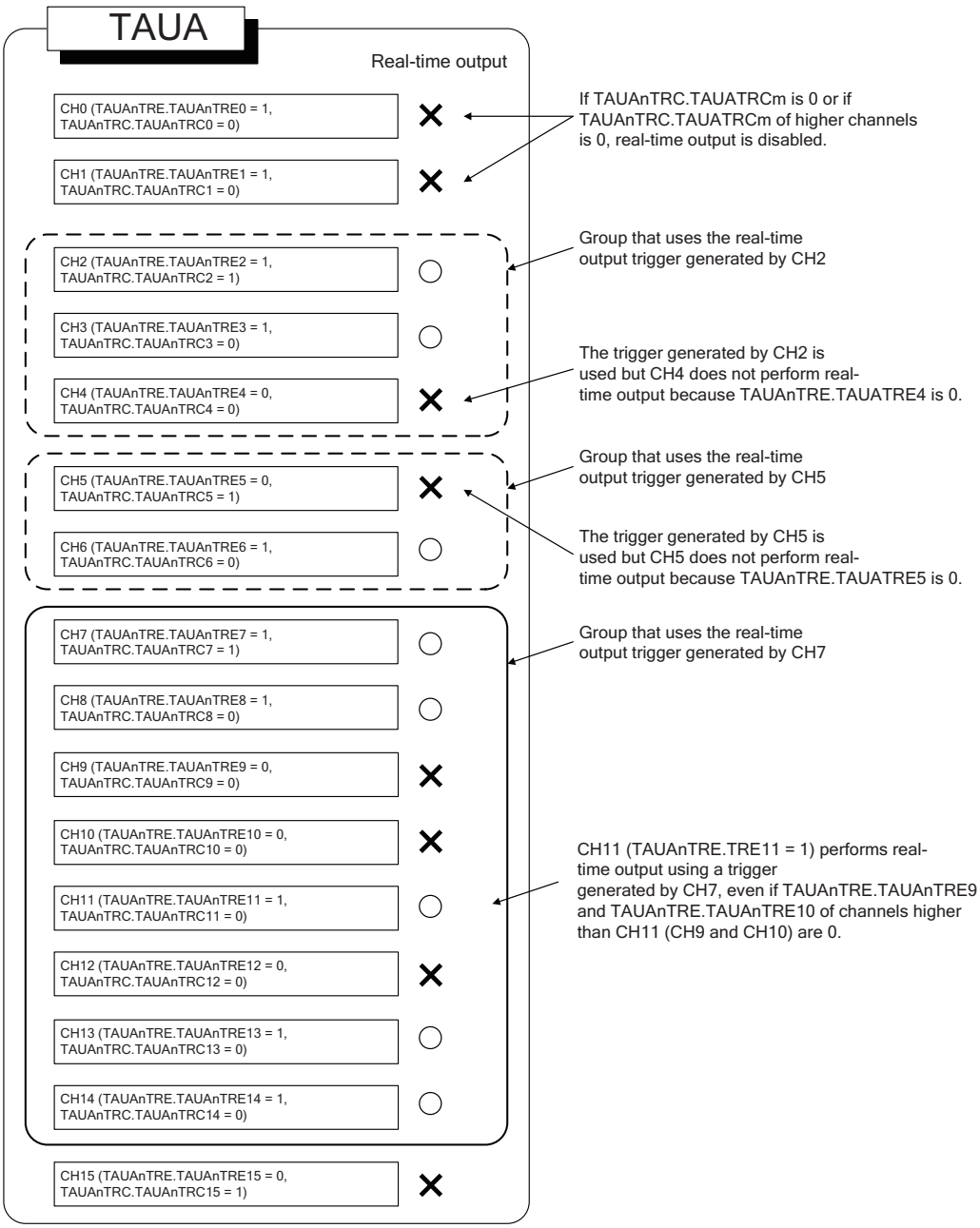


Figure 15-11 Real-time output

(3) Independent channel output mode 2

Set/reset conditions In this output mode, TAUAnTTOUTm is set when INTTAUAnIm occurs upon count start and reset when INTTAUAnIm occurs due to a match between TAUAnCNTm and TAUAnCDRm.

Prerequisites None, other than those in *Table 15-12 “Channel output modes” on page 628*.

15.9.3 Channel output modes controlled synchronously by TAUAn signals

This chapter lists the channel output modes that are controlled synchronously by TAUAn signals. The control bits used to specify a mode are listed in *Table 15-12 "Channel output modes" on page 628*.

(1) Synchronous channel output mode 1

Set/reset conditions

In this output mode, INTTAUAnIm of the master channel serves as the set signal and INTTAUAnIm of the slave channel as the reset signal. If INTTAUAnIm of the master channel and INTTAUAnIm of the slave channel are generated at the same time, INTTAUAnIm of the slave channel (reset signal) has priority over INTTAUAnIm (set signal) of the master channel, i.e. the master channel is ignored.

Prerequisites

None, other than those in *Table 15-12 "Channel output modes" on page 628*.

(2) Synchronous channel output mode 1 with non-complementary modulation output

Set/reset conditions

In this output mode, TAUAnTTOUTm outputs the result of an AND operation between the PWM output of a channel and the real-time output bit (TAUAnTRO.TAUAnTROm).

The phase period to which the dead time is added is specified using the TAUAnTDL.TAUAnTDLm bit; for positive phase set TAUAnTDL.TAUAnTDLm = 0 and for negative phase set TAUAnTDL.TAUAnTDLm = 1.

Prerequisites

A set of at least three channels is required to generate the PWM output. The master channel and slave 1 generate the period, and slave 2 generates the duty cycle. In typical applications, a further 5 slave channels are also used that operate in the same manner as slave 2.

Only the PWM output and the real-time output bit of the same channel can be combined.

TAUAnTRO.TAUAnTROm, TAUAnTME.TAUAnTMEm, and TAUAnTDL.TAUAnTDLm can only be changed during count operation.

- If TAUAnTME.TAUAnTMEm is changed, the new value of TAUAnTME.TAUAnTMEm is applied upon detection of INTTAUAnIm on the specified channel.
- If TAUAnTME.TAUAnTMEm and TAUAnTDL.TAUAnTDLm are changed, the new values are applied upon detection of INTTAUAnIm on the master channel.

(3) Synchronous channel output mode 2

In this output mode, the operation mode must be set to up/down count mode. The result is a triangle PWM wave at TAUAnTTOUTm. For details, see *15.25.1 "Triangle PWM output function" on page 826*.

Set/reset conditions TAUAnCNTm of the slave channel counts down and up alternatively. When it passes 0001_H it generates an interrupt, causing TAUAnTTOUTm to toggle.

Prerequisites A set of two channels is required to generate the triangle PWM output. TAUAnTTOUTm must be set to 0 before the function starts.

(4) Synchronous channel output mode 2 with dead time output

In this output mode, a dead time delay is added to TAUAnTTOUTm. The set/reset conditions are shown in the following figure.

Set/reset conditions

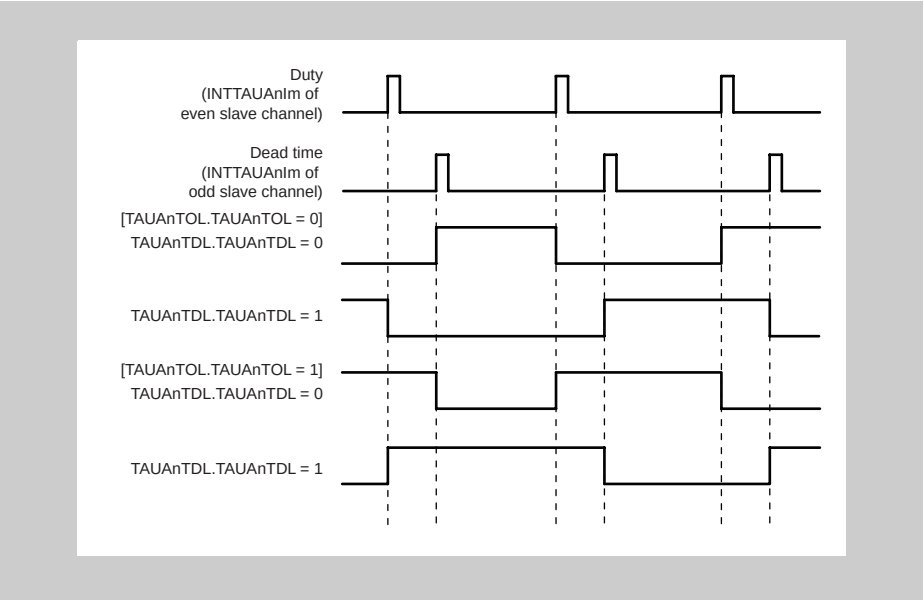


Figure 15-12 Set/reset conditions for synchronous channel output mode 2 with dead time output

The edge to which the dead time is added is specified using the TAUAnTDL.TAUAnTDLm bit; for rising edge set TAUAnTDL.TAUAnTDLm = 0 and for falling edge set TAUAnTDL.TAUAnTDLm = 1.

Prerequisites Dead time control requires a set of three channels, each operating in the following modes:

- One master channel

  The master channel must be set to interval timer mode

- One even slave channel

  The even slave channel must be set to up/down count mode

- One odd slave channel (even channel + 1)

  The odd slave channel must be set to one-count mode

The values of the following bits must be the same for the odd channel and the even channel:

- TAUAnTOE.TAUAnTOEm,
- TAUAnTME.TAUAnTMEm,
- TAUAnTRE.TAUAnTREm,
- TAUAnTOM.TAUAnTOMm,
- TAUAnTOC.TAUAnTOCm,
- TAUAnTDE.TAUAnTDEm, and
- TAUAnTDM.TAUAnTDMm

(5) Synchronous channel output mode 2 with one-phase PWM output

In this output mode, a dead time delay is added to TAUAnTTOUTm. The set/reset conditions are shown in the following figure.

Set/reset conditions

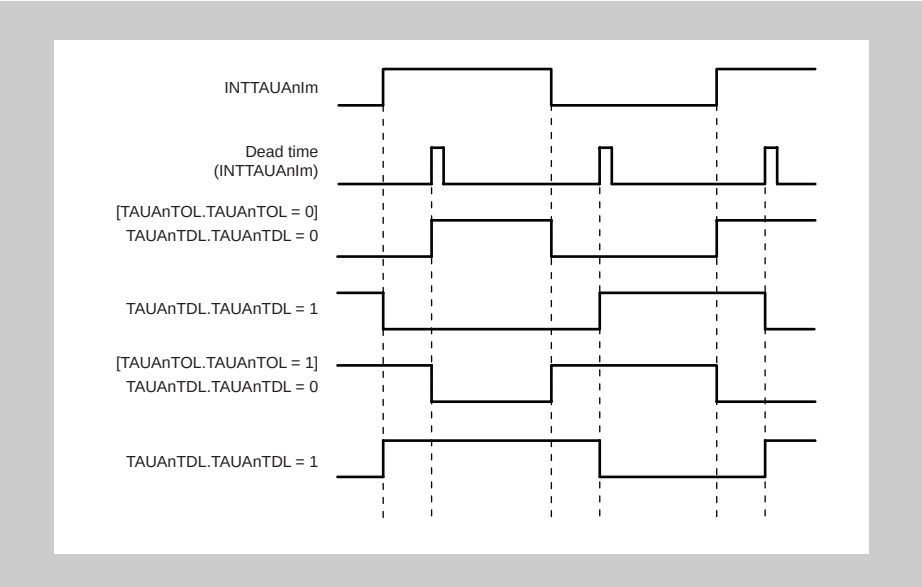


Figure 15-13 Set/reset conditions for synchronous channel output mode 2 with one-phase PWM output

The edge to which the dead time is added is specified using the TAUAnTDL.TAUAnTDLm bit; for rising edge set TAUAnTDL.TAUAnTDLm = 0 and for falling edge set TAUAnTDL.TAUAnTDLm = 1.

Prerequisites One-phase PWM output control requires a set of two channels:

- One even slave channel
- One odd slave channel (even channel + 1)

  The odd slave channel must be set to “one-count mode”

The values of the following bits must be the same for the odd channel and the even channel:

- TAUAnTOE.TAUAnTOEm,
- TAUAnTME.TAUAnTMEm,
- TAUAnTRE.TAUAnTREm,
- TAUAnTOM.TAUAnTOMm,
- TAUAnTOC.TAUAnTOCm,
- TAUAnTDE.TAUAnTDEm, and
- TAUAnTDM.TAUAnTDMm

(6) Synchronous channel output mode 2 with complementary modulation output

Set/reset conditions

In this output mode, TAUAnTTOUTm outputs a PWM signal, a high signal, or a low signal depending on the value of the real-time output bit (TAUAnTRO.TAUAnTROm), the modulation output bit (TAUAnTME.TAUAnTMEm), and the output level bit (TAUAnTOL.TAUAnTOLm) of a pair of slave channels.

For details, see *15.26.3 “Complementary modulation output function” on page 881*.

Prerequisites

A set of at least four channels is required for this mode. The master channel and slave 1 generate the period, slave channel 2 generates the duty cycle, and slave 3 generates the dead time. Slave 2 and 3 are a pair. In typical applications, a further 4 channels are also used that operate in the same manner as slaves 2 and 3 respectively.

TAUAnTRO.TAUAnTROm, TAUAnTME.TAUAnTMEm, and TAUAnTDL.TAUAnTDLm can only be changed during count operation.

- If TAUAnTME.TAUAnTMEm is changed during operation, the new TAUAnTME.TAUAnTMEm value is applied upon detection of INTTAUAnIm at the specified channel.
- If TAUAnTME.TAUAnTMEm and TAUAnTDL.TAUAnTDLm are changed, the new values are applied upon detection of INTTAUAnIm on an even slave channel.

(7) Synchronous channel output mode 2 with non-complementary modulation output

The difference to "synchronous channel output mode 1 with non-complementary modulation output" is the PWM wave shape.

It is a rectangular wave with mode 1 while it is a triangular wave with mode 2.

15.10 Count Start Timing in Each Operating Mode

This section describes the timing for starting counting after the TAUAnTS.TAUAnTSm bit is set to 1, for each operation mode.

The value of the data register and whether an interrupt is generated depend on the individual mode and corresponding register settings.

Caution The timing for starting counting in this section is only for reference. The timing for starting counting actually varies according to the count clock timing.

15.10.1 Interval timer mode, judge mode, capture mode, and up/down count mode

The counter starts counting at the start of the next count clock cycle after TAUAnTS.TAUAnTSm is set to 1. The value of data register is also loaded when the counter starts.

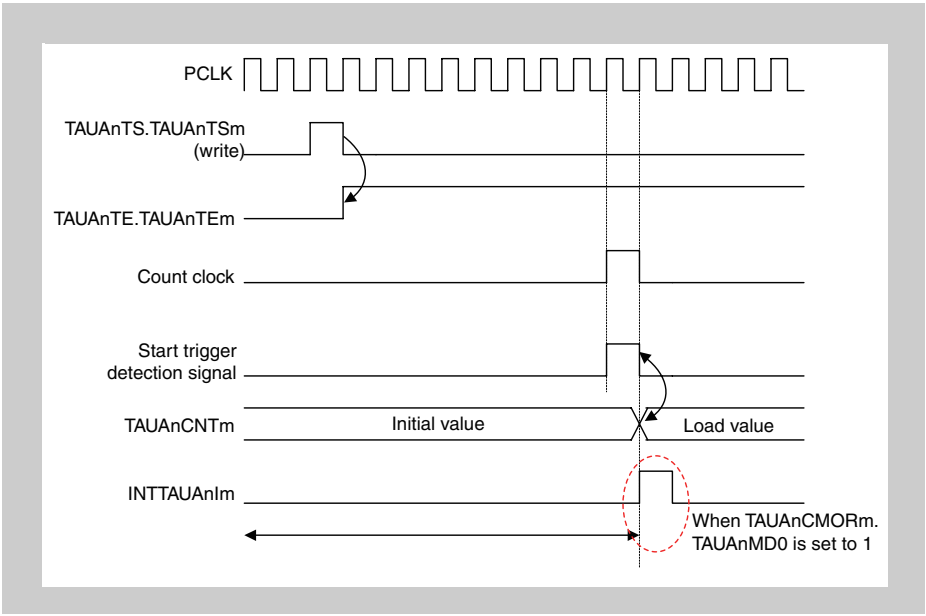


Figure 15-14 Count start timing in interval timer mode, judge mode, capture mode, and up/down count mode

15.10.2 Event count mode

The value of the data register is loaded as soon as TAUAnTS.TAUAnTSm is set to 1. The counter also starts immediately. The value of the data register increments at the start of each subsequent count clock cycle.

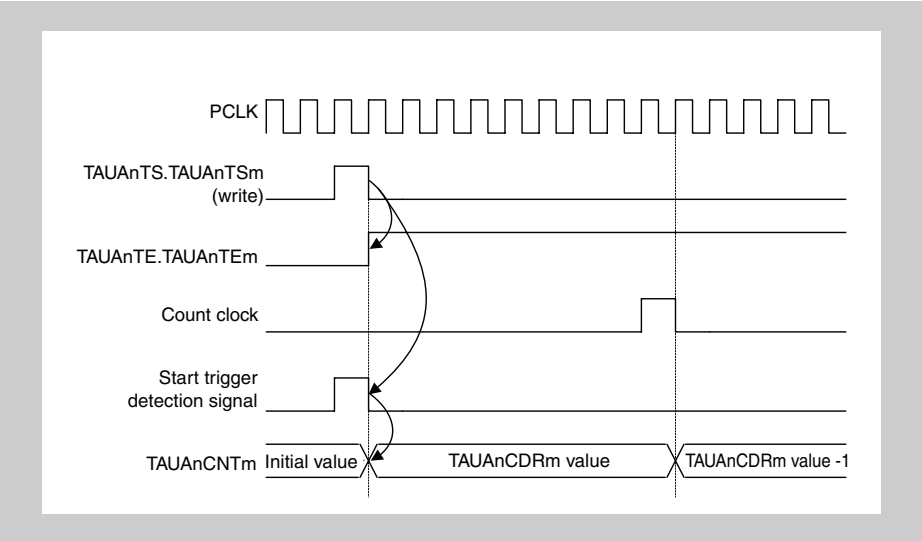


Figure 15-15 Count start timing in event count mode

15.10.3 Other operating modes

In other operating modes, the count clock cycles are ignored with regard to starting the counter. The counter is only triggered by detection of a valid TAUAnTTINm edge. The value of data register is also loaded at the point the counter starts. Nevertheless, the count clock cycles determine the frequency with which all operations take place.

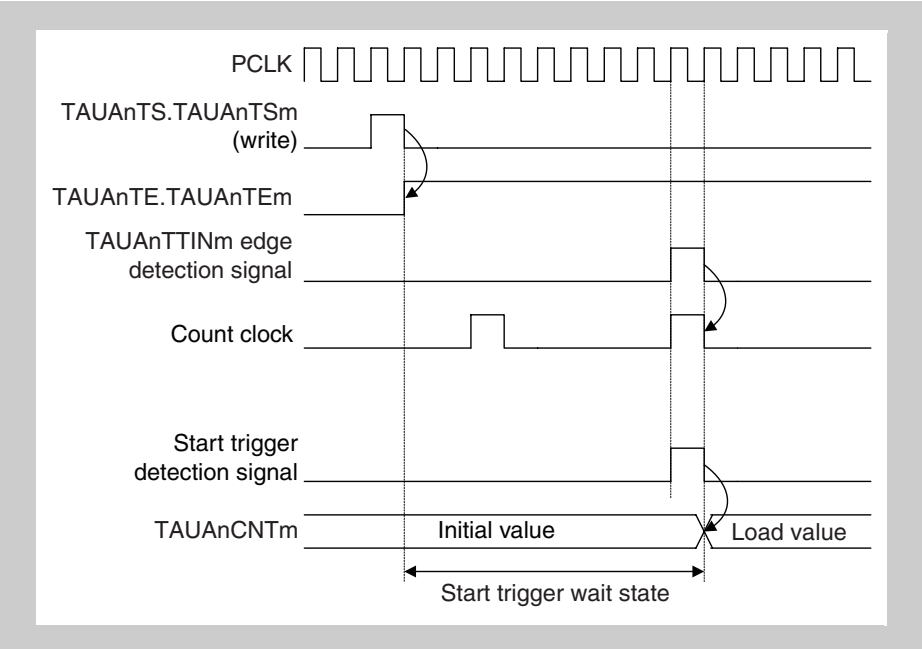


Figure 15-16 Count start timing in other operating modes

15.11 TAUAnTTOUTm Output and INTTAUAnIm Generation When Counter Starts or Restarts

When the counter starts, it is possible to specify whether an INTTAUAnIm is generated using the TAUAnCMORn.TAUAnMD0 bit. The effect of the bit depends on the selected mode, as shown in the table below. The effects of INTTAUAnIm on TAUAnTTOUTm depend on the selected channel operation function.

Table 15-13 Effect of TAUAnCMORm.TAUAnMD0 bit on generation of INTTAUAnIm when counter is triggered

Mode	TAUAnCMORm. TAUAnMD0 bit	INTTAUAnIm generated when counter starts
Interval timer mode Capture mode Count capture mode	0	No
	1	Yes
Capture & one-count mode Capture & gate count mode Event count mode Up/down count mode	0	No
One-count mode Gate count mode	0/1	No, regardless of setting of TAUAnCMORn.TAUAnMD0 bit.
Pulse one-count mode		Yes, regardless of setting of TAUAnCMORn.TAUAnMD0 bit.

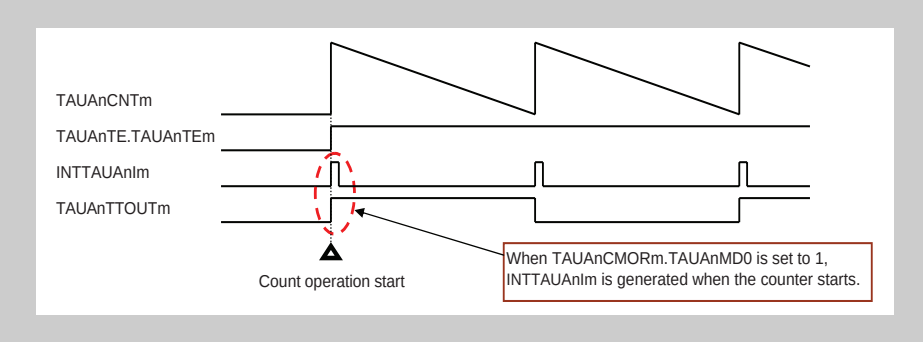


Figure 15-17 INTTAUAnIm generated when counter starts

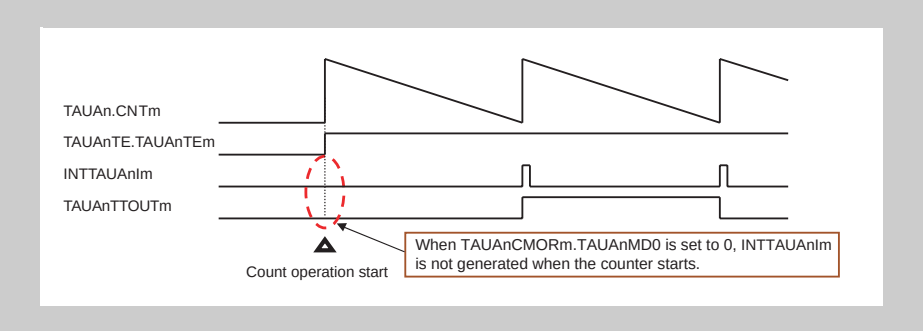


Figure 15-18 INTTAUAnIm not generated when counter starts

15.12 Interrupt Generation upon Overflow

Certain independent functions that count up, overflow without generating an interrupt when they reach $FFFF_H$. This section describes how it is possible to generate an interrupt, by combining a channel operating in one of these modes with a channel in a different operation mode which counts down.

The appropriate operation mode for the second channel depends on the operation mode of the first channel. Nevertheless, the principle is the same for all combinations:

- Find a operation mode for the second channel that counts down in such a manner, that it reaches 0000_H at the same time as the first channel overflows (TAUAnCNTm = $FFFF_H$).
- Set TAUAnCDRm of the second channel to $FFFF_H$
- The two channels must count at the same speed (i.e. they must have the same count clock)
- Both channels are triggered by the same TAUAnTTINm input

Result: The down-counter of the second channel reaches 0000_H at exactly the same time as the up-counter of the first channel overflows (TAUAnCNTm = $FFFF_H$). Thus the second channel generates the desired interrupt.

The following sections list the operating modes that count down that are required to match specific operating modes that count up, as well as example timing diagrams.

15.12.1 Capture mode

Applies to • TAUAnTTINm input pulse interval measurement function

Combine with Interval timer mode

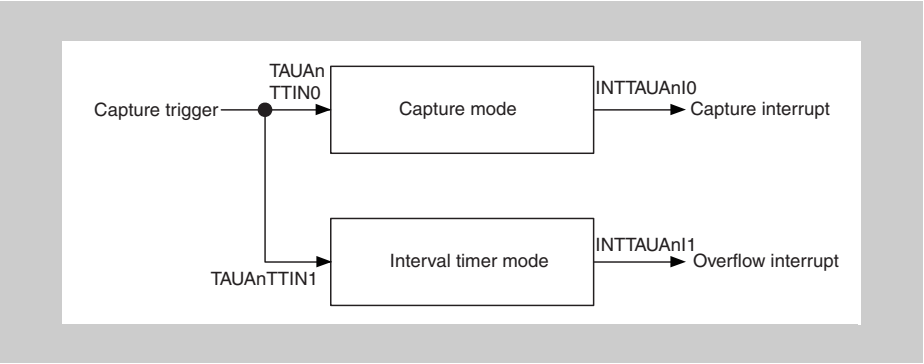


Figure 15-19 Combination of capture mode and interval timer mode

Timing diagram

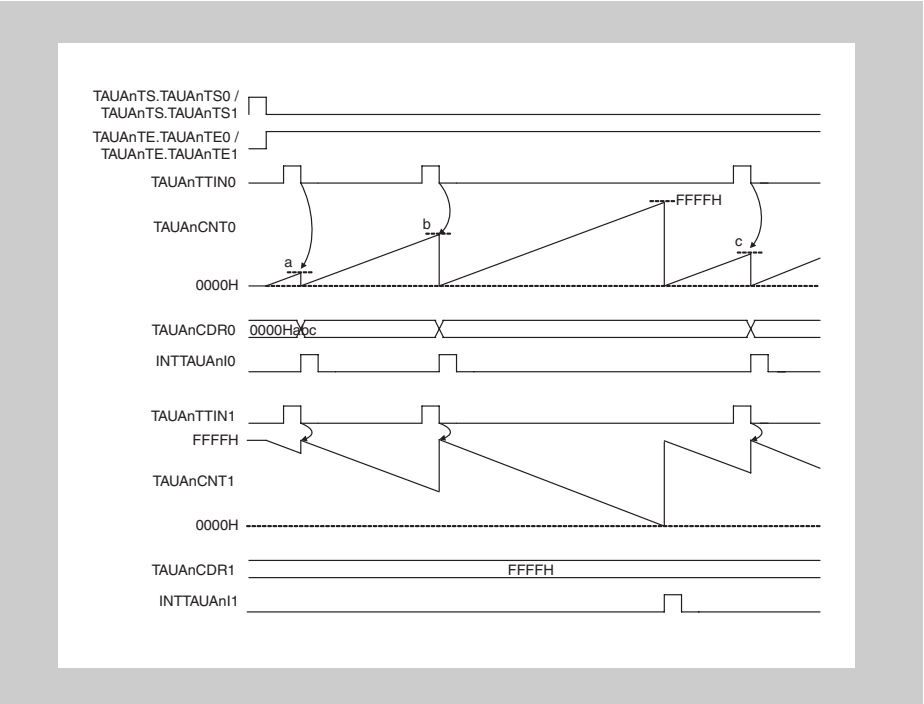


Figure 15-20 Interrupt generation via combination of capture mode and interval timer mode

15.12.2 Capture & one-count mode

Applies to
- TAUAnTTINm input signal width measurement function

Combine with
One-count mode

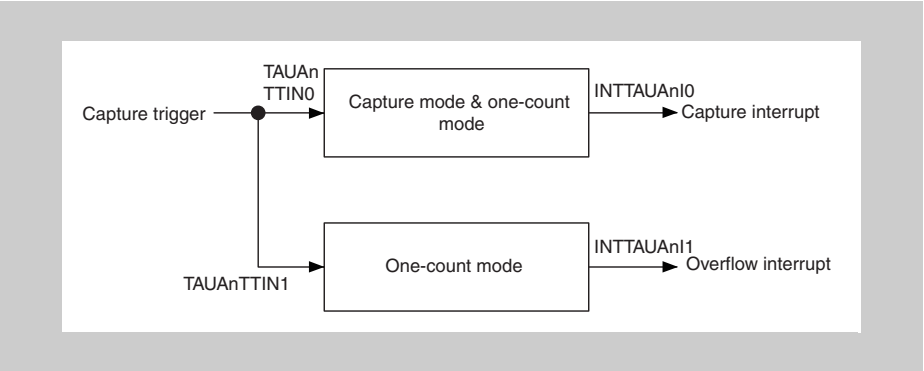


Figure 15-21 Combination of capture & one-count mode and one-count mode

Timing diagram

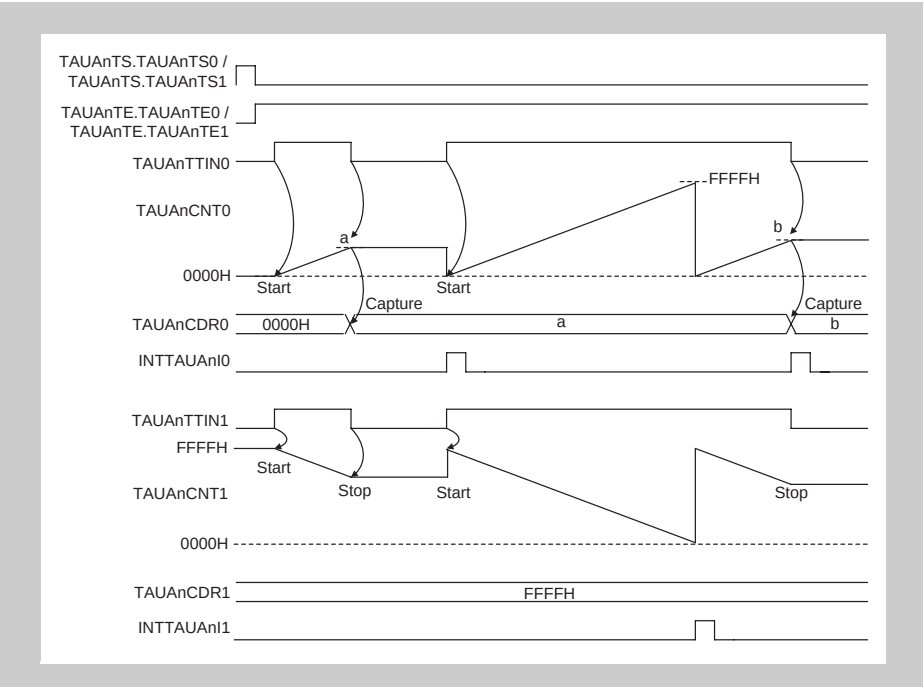


Figure 15-22 Interrupt generation via combination of capture & one-count mode and one-count mode

15.12.3 Count capture mode

Applies to
- TAUAnTTINm input position detection function

Combine with
Interval timer mode

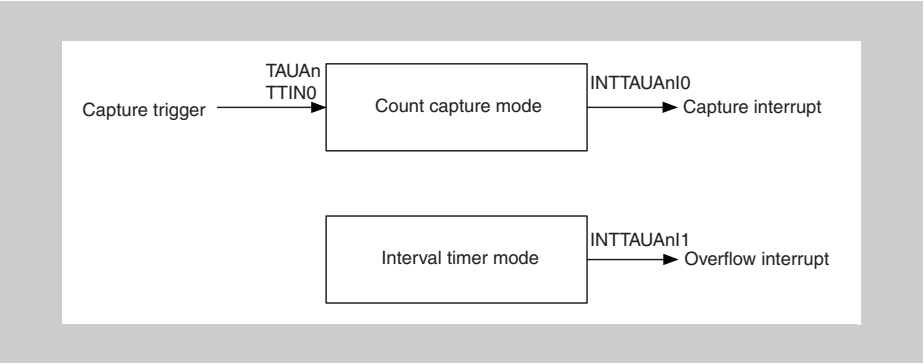


Figure 15-23 Combination of count capture mode and interval timer mode

Timing diagram

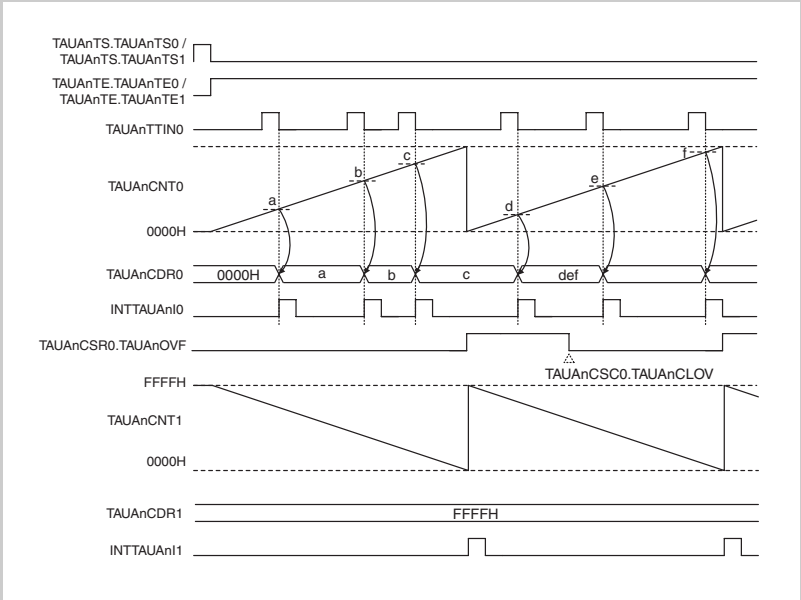


Figure 15-24 Interrupt generation via combination of count capture mode and interval timer mode

In the above timing diagram, TAUAnCSRm.TAUAnOVF is set to 1 when TAUAnCNTm overflows.

TAUAnCSRm.TAUAnOVF is cleared by writing 1 to TAUAnCSCm.TAUAnCLOV.

15.12.4 Capture & gate count mode

Applies to • TAUAnTTINm input period count detection function

Combine with Gate count mode

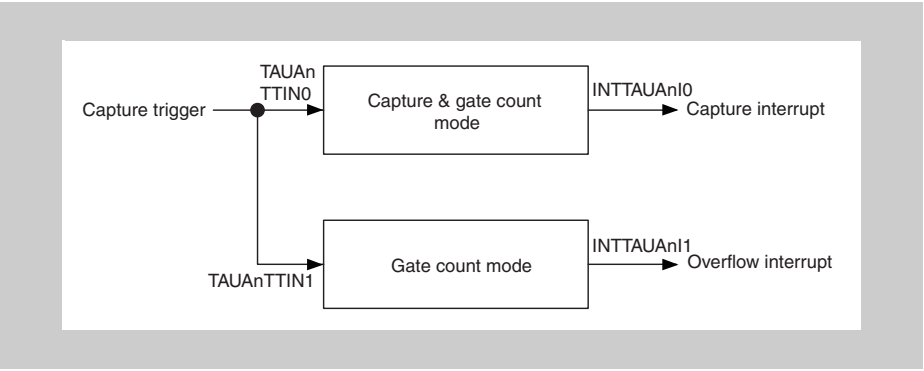


Figure 15-25 Combination of capture & gate count mode and gate count mode

Timing diagram

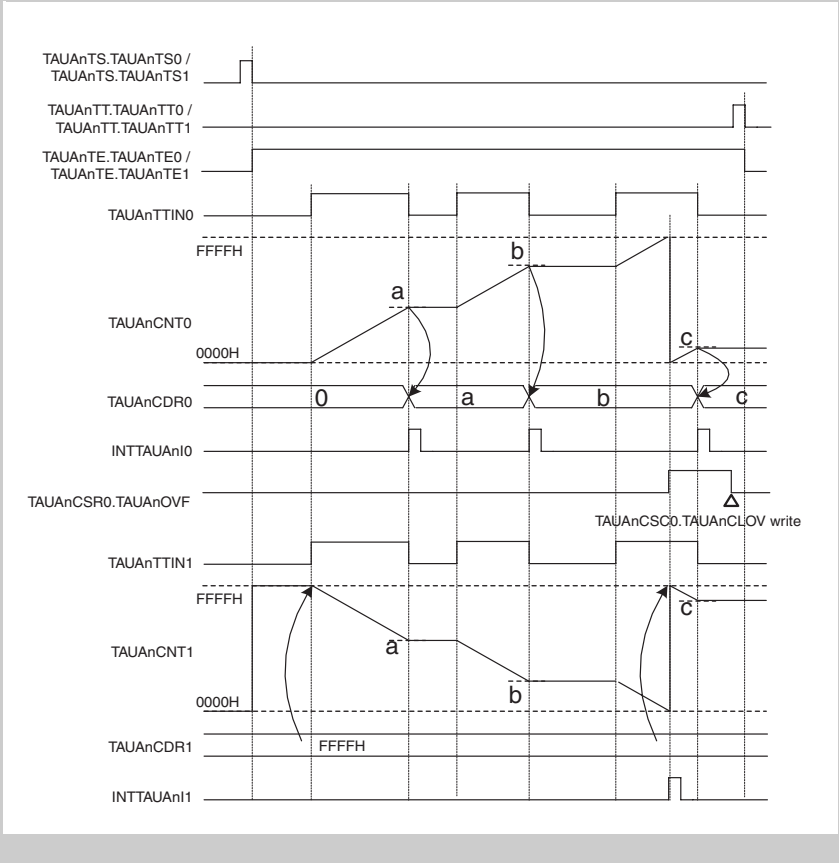


Figure 15-26 Interrupt generation via combination of capture & gate count mode and gate count mode

In the above timing diagram, TAUAnCSRm.TAUAnOVF is set to 1 when TAUAnCNTm overflows. TAUAnCSRm.TAUAnOVF is cleared by writing 1 to TAUAnCSCm.TAUAnCLOV.

15.13 TAUAnTTINm Edge Detection

Edge detection is based on the operation clock. This means that an edge can only be detected at the next rising edge of the operation clock. This can lead to a maximum delay of one operation clock cycle.

The following figure shows an edge detection timing example.

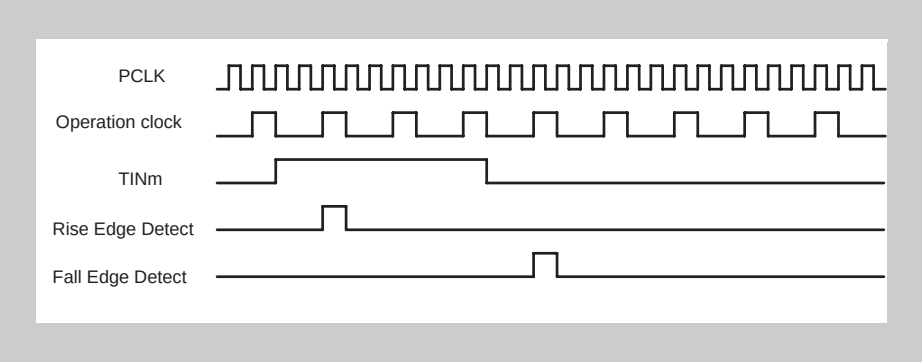


Figure 15-27 Basic edge detection timing

Figure 15-27 "Basic edge detection timing" shows the approximate operation timing. In actuality, there is delay due to the noise filter and synchronizer between the TAUAnIm pin and TAUAn.

15.14 Assigning DMA Window Addresses

DMA (direct memory access) can be used to store values in the TAUAnDWR registers, for example the current values of the TAUAnCDRm register and the TAUAnTOL register.

For example, the following figure shows how to rewrite the period register, duty registers, and TAUAnTOL register to the TAUAnDWR registers.

1. Specify the addresses of the selected registers (TAUAnCDR0, TAUAnCDR2, TAUAnCDR4, TAUAnCDR6, TAUAnCDR9, TAUAnCDR13, TAUAnTOL.TAUAnTOLm, and TAUAnRDT.TAUAnRDTm) to the TAUAnDAS registers.
2. The TAUAnDMA window address function then loads the values of the selected registers to the corresponding TAUAnDWR registers.

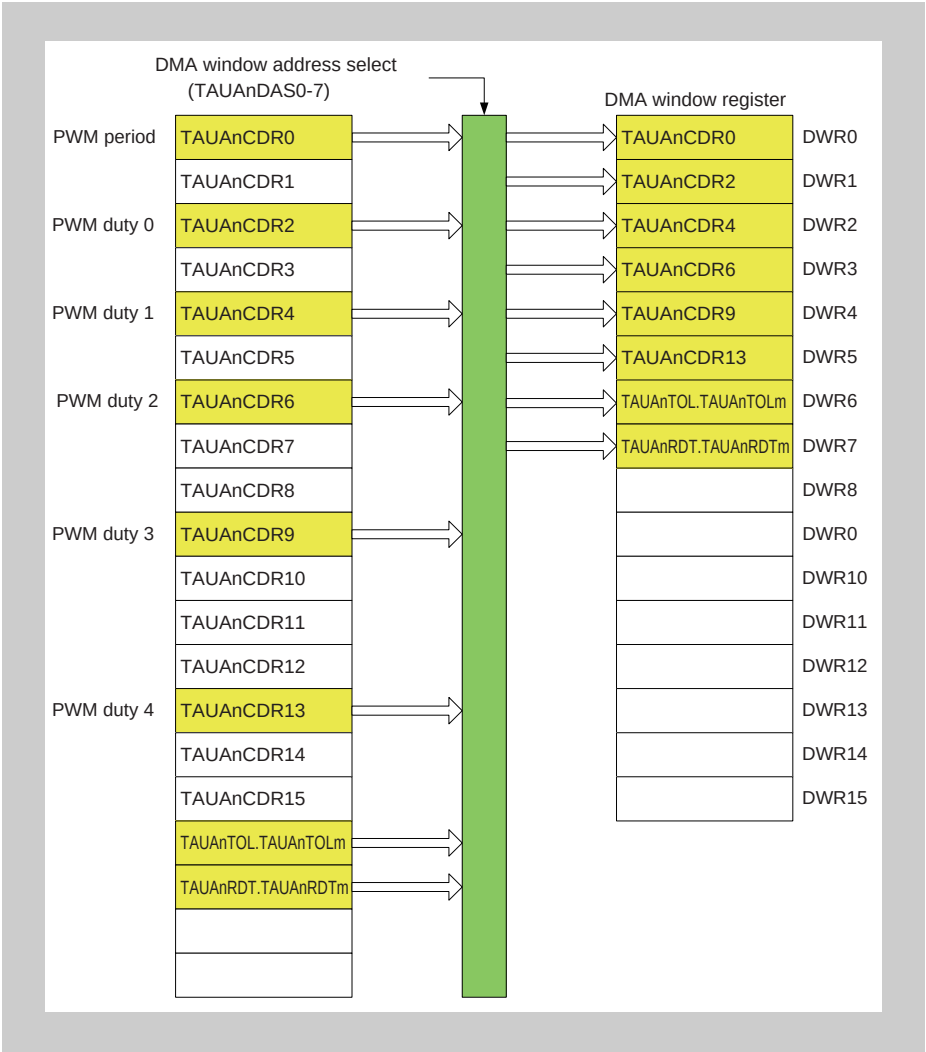


Figure 15-28 Assigning DMA window addresses

Note When using the simultaneous rewrite function for a data register, it is recommended to allocate the TAUAnRDT register at the end of the specified area.

15.15 Independent Channel Operation Functions

The following sections list the independent channel operation functions provided by TAUA. For a general overview of independent channel operation, see *15.4 “Functional Description” on page 608*.

15.16 Independent Channel Interrupt Functions

This chapter describes functions that generate interrupts at regular intervals or with a specified delay.

- *15.16.1 “Interval timer function”*
- *15.16.2 “TAUAnTTINm input interval timer function”*
- *15.16.3 “Delay count function”*
- *15.16.4 “One-pulse output function”*

15.16.1 Interval timer function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUAnIm) at regular intervals. When an interrupt is generated, the TAUAnTTOUTm signal toggles, resulting in a square wave.

Prerequisites
- The operation mode must be set to interval timer mode. See *Table 15-14 "TAUAnCMORm settings for interval timer function" on page 650*.
- The channel output mode must be set to independent channel output mode 1. See *15.9 "Channel Output Modes" on page 627*.

Description The counter is started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. The current value of TAUAnCDRm is loaded to TAUAnCNTm and the counter starts to count down from this value.

When the counter reaches 0000_H, INTTAUAnIm is generated and the TAUAnTTOUTm signal toggles. Next, TAUAnCNTm loads the value of TAUAnCDRm, and then subsequently continues operation.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1, which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm stop but retain their values. The counter can be reset by setting TAUAnTS.TAUAnTSm to 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUAnTS.TAUAnTSm to 1 during operation.

Conditions If the TAUAnCMORm.TAUAnMD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUAnTTOUTm does not toggle. This results in an inverted TAUAnTTOUTm signal compared to when TAUAnCMORm.TAUAnMD0 is set to 1. For details, see *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation When Counter Starts or Restarts" on page 639*.

(2) Equations

INTTAUAnIm cycle = count clock cycle x (TAUAnCDRm + 1)

TAUAnTTOUTm square wave cycle = count clock cycle x (TAUAnCDRm + 1) x 2

(3) Block diagram and general timing diagram

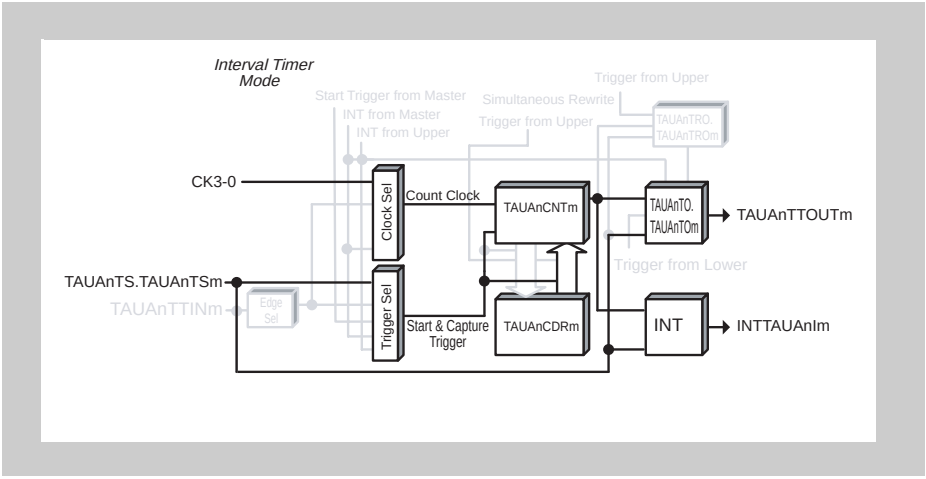


Figure 15-29 Block diagram for interval timer function

The following settings apply to the general timing diagram:

- INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)

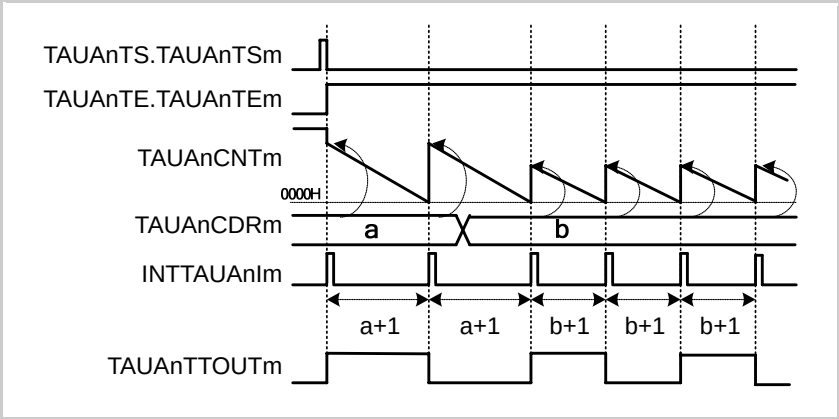


Figure 15-30 General timing diagram for interval timer function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-14 TAUAnCMORm settings for interval timer function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start or restart 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-15 TAUAnCMURm settings for interval timer function

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Table 15-16 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

Note The channel output mode can also be set to channel output mode controlled by software by setting TAUAnTOE.TAUAnTOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details, see *15-12 “Channel output modes” on page 628*.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the interval timer function. Therefore, these registers must be set to 0.

Table 15-17 Simultaneous rewrite settings for interval timer function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(5) Operating procedure for interval timer function

Table 15-18 Operating procedure for interval timer function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-14 "TAUAnCMORm settings for interval timer function" on page 650* and *Table 15-15 "TAUAnCMURm settings for interval timer function" on page 650*. Set the value of the TAUAnCDRm register. Set the channel output mode by setting the control bits as described in *Table 15-16 "Control bit settings for independent channel output mode 1" on page 651*.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. When TAUAnCMORm.TAUAnMD0 = 1, INTTAUAnIm is generated and TAUAnTTOUTm toggles.
During operation	The TAUAnCDRm register value can be changed at any time. The TAUAnCNTm register can be read at all times.	TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value, and then continues count operation. • INTTAUAnIm is generated and TAUAnTTOUTm toggles.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

(a) TAUAnCDRm = 0000_H, count clock = PCLK/2

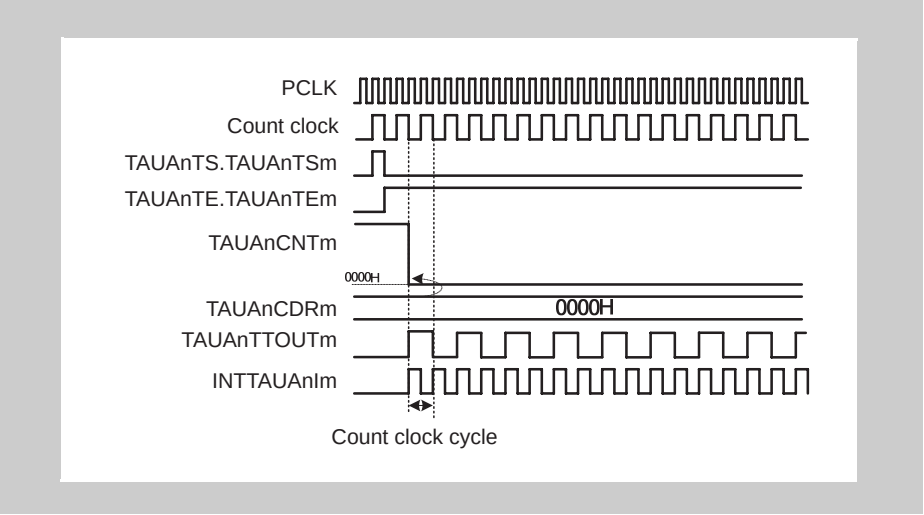


Figure 15-31 TAUAnCDRm = 0000_H, count clock = PCLK/2

- If TAUAnCDRm = 0000_H and the count clock = PCLK/2[1], the TAUAnCDRm value is loaded to TAUAnCNTm every count clock, meaning that TAUAnCNTm is always 0000_H.
- INTTAUAnIm is generated every count clock, resulting in TAUAnTTOUTm toggling every count clock.

(b) TAUAnCDRm = 0000_H, count clock = PCLK

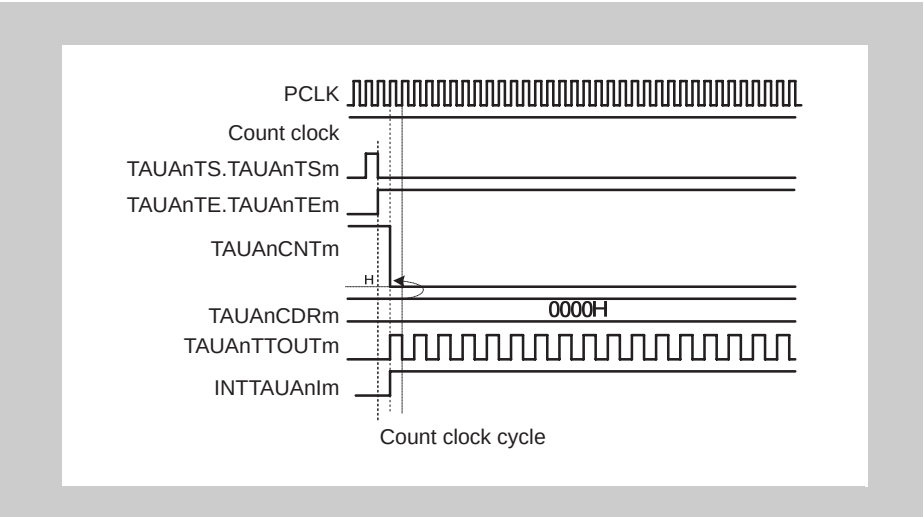


Figure 15-32 TAUAnCDRm = 0000_H, count clock = PCLK

- If TAUAnCDRm = 0000_H and the count clock = PCLK, the TAUAnCDRm value is loaded to TAUAnCNTm every PCLK clock, meaning that TAUAnCNTm is always 0000_H.
- INTTAUAnIm is generated continuously, resulting in TAUAnTTOUTm toggling every PCLK clock.

(c) Operation stop and restart

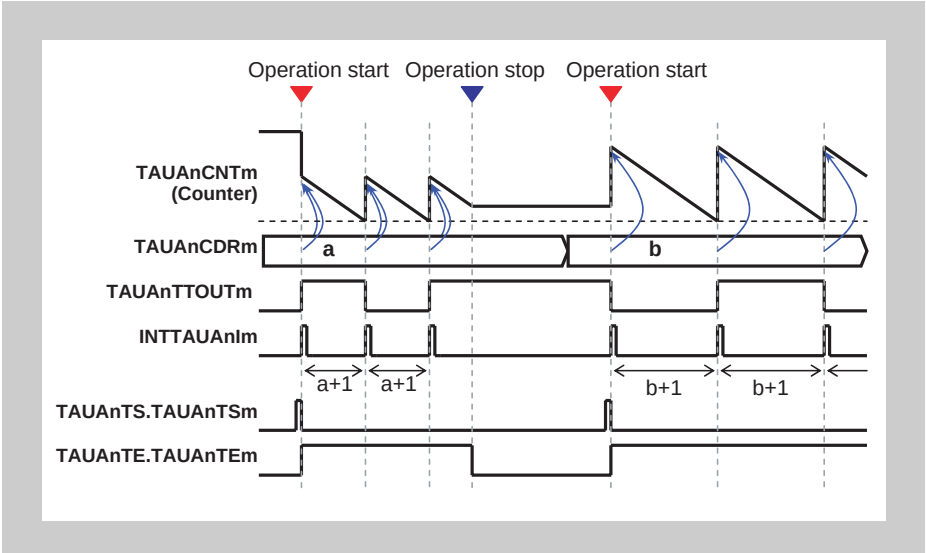


Figure 15-33 Operation stop and restart, TAUAnCMORm.TAUAnMD0 = 1

- The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1, which in turn sets TAUAnTE.TAUAnTEm to 0.
- TAUAnCNTm and TAUAnTTOUTm stop but retain their values.
- The counter can be restarted by setting TAUAnTS.TAUAnTSm to 1.

(d) Forced restart

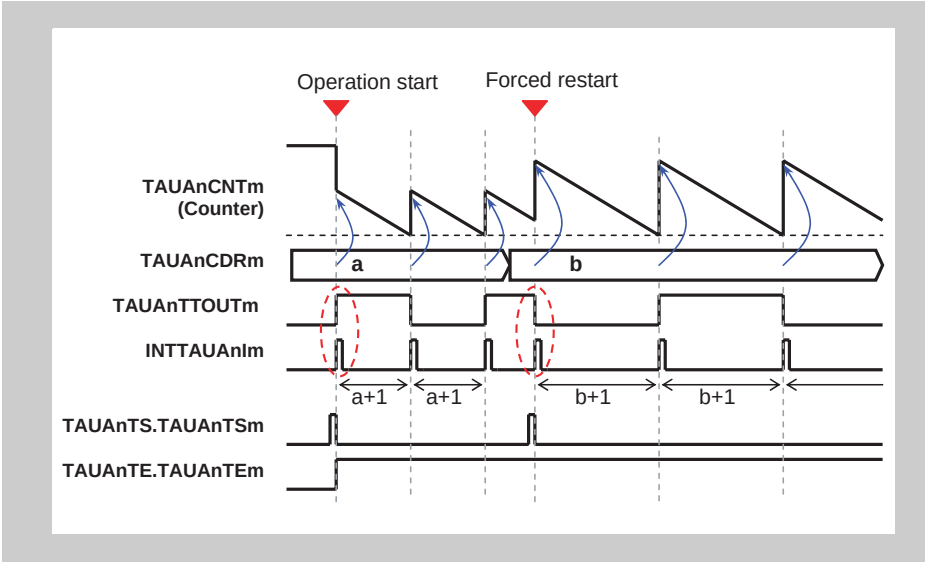


Figure 15-34 Forced restart operation, TAUAnCMORm.TAUAnMD0 = 1

- The counter can be forcibly restarted (without stopping it first) by setting TAUAnTS.TAUAnTSm to 1 during operation.
- If the TAUAnCMORm.TAUAnMD0 bit is set to 1, the first interrupt after a start or restart is generated.

15.16.2 TAUAnTTINm input interval timer function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUAnIm) at regular intervals or when a valid TAUAnTTINm input edge is detected. When an interrupt is generated, the TAUAnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to interval timer mode. See *Table 15-19 "TAUAnCMORm settings for TAUAnTTINm Input interval timer function" on page 657*.
- The channel output mode must be set to independent channel output mode 1. See *15.9 "Channel Output Modes" on page 627*.

Description This function operates in an identical manner to the interval timer function (see *15.16.1 "Interval timer function" on page 648*), except that this function is restarted by a valid TAUAnTTINm input edge. The type of edge used as the trigger is specified using the TAUAnCMURm.TAUAnTIS[1:0] bits. Either rising edge, falling edge, or rising and falling edge can be selected.

(2) Equations

INTTAUAnIm cycle = count clock cycle x (TAUAnCDRm + 1)

TAUAnTTOUTm square wave cycle = count clock cycle x (TAUAnCDRm + 1) x 2

(3) Block diagram and general timing diagram

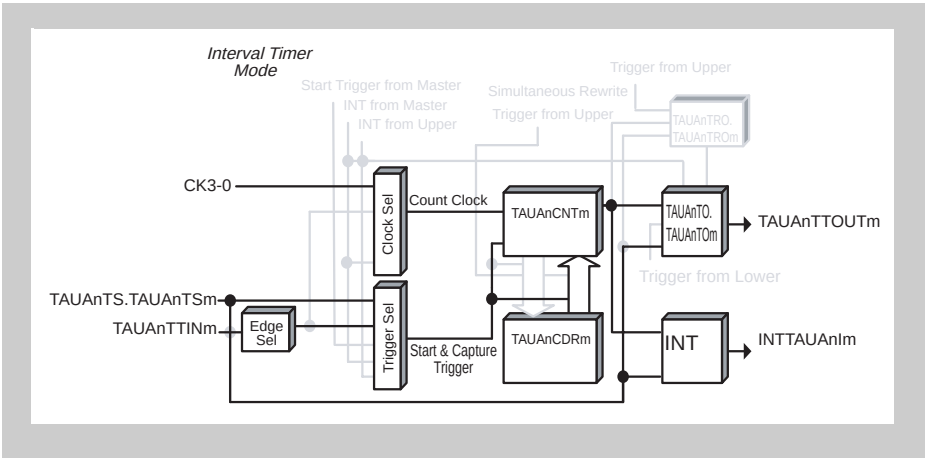


Figure 15-35 Block diagram for TAUAnTTINm Input interval timer function

The following settings apply to the general timing diagram:

- INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)
- Rising edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 01_B)

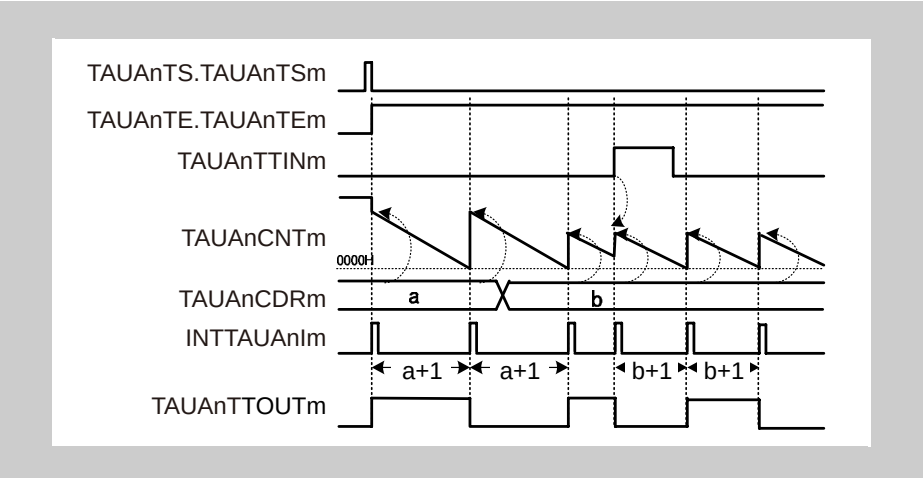


Figure 15-36 General timing diagram for TAUAnTTINm Input interval timer function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-19 TAUAnCMORm settings for TAUAnTTINm Input interval timer function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the external start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-20 TAUAnCMURm settings for TAUAnTTINm Input interval timer function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Table 15-21 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

Note The channel output mode can also be set to channel output mode controlled by software by setting TAUAnTOE.TAUAnTOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details, see *15-12 “Channel output modes” on page 628*.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm input interval timer function. Therefore, these registers must be set to 0.

Table 15-22 Simultaneous rewrite settings for TAUAnTTINm input interval timer function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(5) Operating procedure for TAUAnTTINm input interval timer function

Table 15-23 Operating procedure for TAUAnTTINm input interval timer function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-19 "TAUAnCMORm settings for TAUAnTTINm Input interval timer function" on page 657* and *Table 15-20 "TAUAnCMURm settings for TAUAnTTINm Input interval timer function" on page 657*. Set the value of the TAUAnCDRm register. Set the channel output mode by setting the control bits as described in *Table 15-21 "Control bit settings for independent channel output mode 1" on page 658*.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTAUAnTSm to 1. TAUAnTS.TAUAnTAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. When TAUAnCMORm.TAUAnMD0 = 1, INTTAUAnIm is generated and TAUAnTTOUTm toggles.
During operation	The values of the TAUAnCMURm.TAUAnTAUAnTIS[1:0] bits and the TAUAnCDRm register can be changed at any time. The TAUAnCNTm register can be read at all times. Detection of TAUAnTTINm edge	TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value, and then continues count operation. • INTTAUAnIm is generated and TAUAnTTOUTm toggles. When a TAUAnTTINm input valid edge is detected during count operation, TAUAnCNTm reloads the TAUAnCDRm value and continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

The timing diagrams in *15.16.1 "Interval timer function" on page 648* also apply, except for this function the counter can also be restarted by a valid TAUAnTTINm input edge.

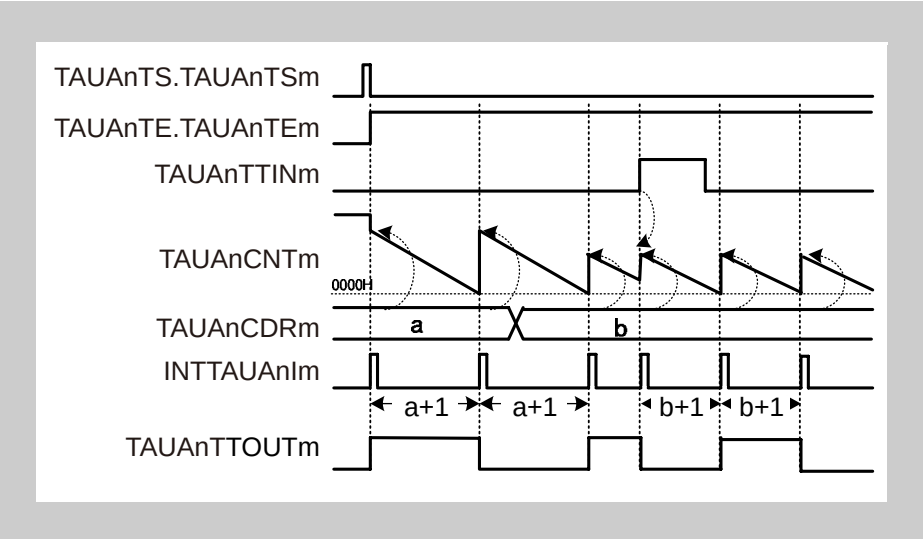


Figure 15-37 Counter triggered by rising TAUAnTTINm input edge (TAUAnCMURm.TAUAnTIS[1:0] = 01_B), TAUAnCMORm.TAUAnMD0 = 1

- If a valid TAUAnTTINm input edge is detected, an interrupt is generated which causes TAUAnTTOUTm to toggle. In this example, the valid edge is a rising edge (TAUAnCMURm.TAUAnTIS[1:0] = 01_B).

15.16.3 Delay count function

(1) Overview

Summary This function generates interrupts (INTTAUAnIm), which have a defined delay to the TAUAnTTINm input signal. TAUAnTTINm input signal pulses that occur within the delay period are ignored.

Prerequisites
- The operation mode must be set to one-count mode. See *Table 15-24 "TAUAnCMORm settings for delay count function" on page 663*.
- TAUAnTTOUTm is not used for this function.
- The start trigger must be disabled during counting (TAUAnCMORn.TAUAnMD0 = 0).

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input start edge is detected. The value of TAUAnCDRm is loaded to TAUAnCNTm and the counter starts to count down from the TAUAnCDRm value.

When the counter reaches 0000_H an interrupt is generated. The counter returns to $FFFF_H$ and awaits the next valid TAUAnTTINm input edge.

When the counter is counting down, further TAUAnTTINm input signals are ignored, i.e., the counter does not reset.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUAnCMURm.TAUAnTIS[1:0] bits:
- If TAUAnCMURm.TAUAnTIS[1:0] = 00_B, falling edges trigger the counter.
- If TAUAnCMURm.TAUAnTIS[1:0] = 01_B, rising edges trigger the counter.
- If TAUAnCMURm.TAUAnTIS[1:0] = 10_B, rising and falling edges trigger the counter.

(2) Equations

Delay between TAUAnTTINm and INTTAUAnIm =

count clock cycle × (TAUAnCDRm + 1)

(3) Block diagram and general timing diagram

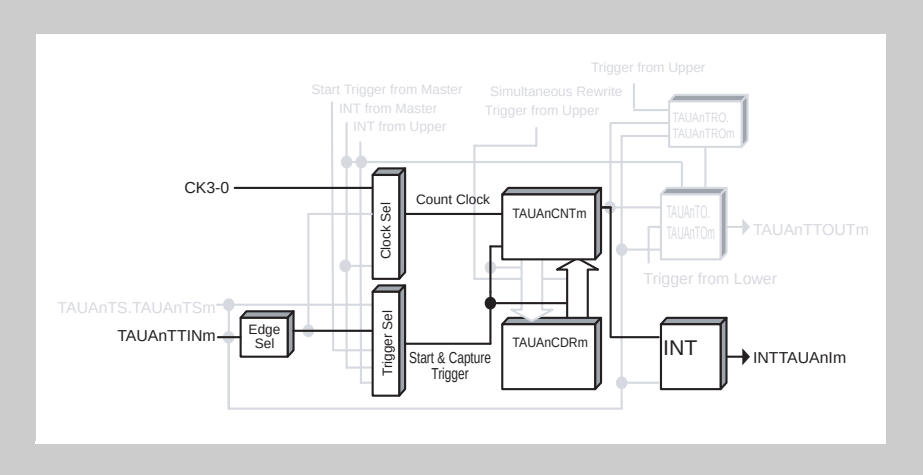


Figure 15-38 Block diagram for delay count function

The following settings apply to the general timing diagram:

- Falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)

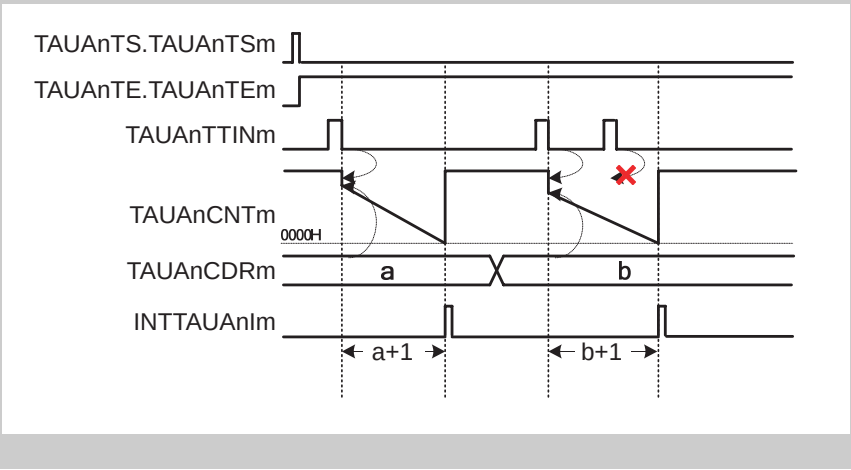


Figure 15-39 General timing diagram for delay count function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAn MD[4:1]				TAUAn MD0

Table 15-24 TAUAnCMORm settings for delay count function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the external start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	0: Disables the start trigger during operation

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-25 TAUAnCMURm settings for delay count function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

<R>
<R>

(c) Channel output mode

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, the channel output mode can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the delay count function. Therefore, these registers must be set to 0.

Table 15-26 Simultaneous rewrite settings for delay count function

Bit name	Setting
TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDMm	
TAUAnRDCm	

(5) Operating procedure for delay count function

Table 15-27 Operating procedure for delay count function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-24 "TAUAnCMORm settings for delay count function" on page 663* and *Table 15-25 "TAUAnCMURm settings for delay count function" on page 663*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TAUAnTEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a start edge is detected, TAUAnCNTm loads the TAUAnCDRm value.
During operation	The values set in the TAUAnCDRm register can be changed at any time. The TAUAnCNTm register can be read at all times.	TAUAnCNTm counts down. When the counter reaches 0000_H: INTTAUAnIm is generated. TAUAnCNTm stops counting, returns to $FFFF_H$, and waits for a trigger. If a trigger occurs while TAUAnCNTm is counting, the trigger is ignored. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its value.

Restart

15.16.4 One-pulse output function

(1) Overview

Summary This function generates an interrupt (INTTAUAnIm) when a valid TAUAnTTINm input edge is detected and also a specific interval later. TAUAnTTINm input signal pulses that occur within the defined interval are ignored. When an interrupt is generated, the TAUAnTTOUTm signal toggles, resulting in a square wave.

Prerequisites
- The operation mode must be set to pulse one-count mode. See *Table 15-28 "TAUAnCMORm settings for one-pulse output function" on page 668*.
- The channel output mode must be set to independent channel output mode 1. See *15.9 "Channel Output Modes" on page 627*.
- Trigger detection must be disabled during counting (TAUAnCMORn.TAUAnMD0 = 0).

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input edge is detected. The value of TAUAnCDRm is loaded to TAUAnCNTm and the counter starts to count down from the TAUAnCDRm value. An interrupt is generated and TAUAnTTOUTm becomes active.

When the counter reaches 0001_H an interrupt is generated and TAUAnTTOUTm becomes inactive. The counter stops at 0000_H and awaits the next valid TAUAnTTINm input edge.

When the counter is counting down, further TAUAnTTINm input signals are ignored, i.e. the counter does not reset.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUAnCMURm.TAUAnTIS[1:0] bits:
- If TAUAnCMURm.TAUAnTIS[1:0] = 00_B, falling edges trigger the counter.
- If TAUAnCMURm.TAUAnTIS[1:0] = 01_B, rising edges trigger the counter.
- If TAUAnCMURm.TAUAnTIS[1:0] = 10_B, rising and falling edges trigger the counter.

(2) Equations

Interval between TAUAnTTINm and INTTAUAnIm = TAUAnTTOUTm (timer output) width = count clock cycle × TAUAnCDRm

(3) Block diagram and general timing diagram

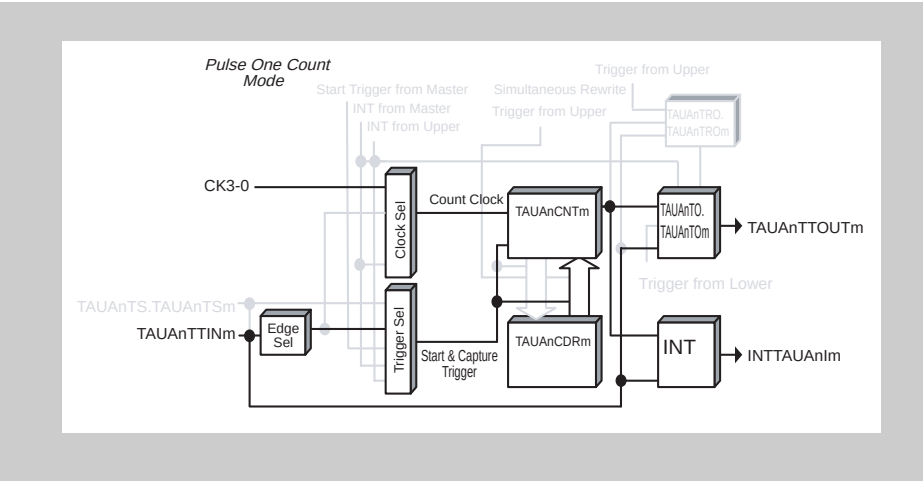


Figure 15-40 Block diagram for one-pulse output function

The following settings apply to the general timing diagram:

- Falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)

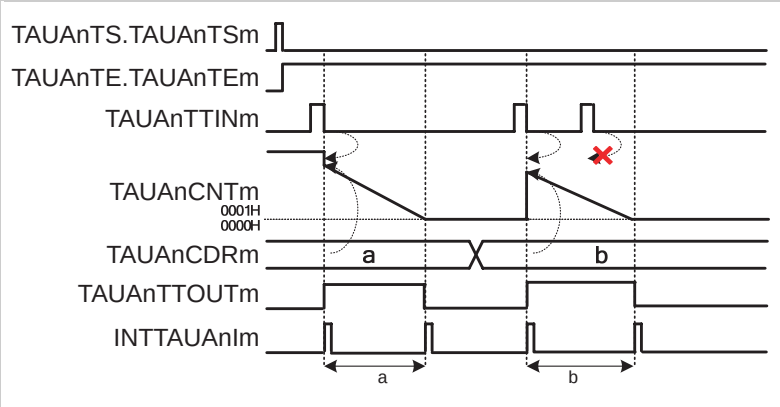


Figure 15-41 General timing diagram for one-pulse output function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAn STS[2:0]			TAUAn COS[1:0]		-	TAUAn MD[4:1]				TAUAn MD0

Table 15-28 TAUAnCMORm settings for one-pulse output function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the external start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	1010: Pulse one-count mode
TAUAnMD0	0: Disables the start trigger during operation

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-29 TAUAnCMURm settings for one-pulse output function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

<R>
<R>

(c) Channel output mode

Table 15-30 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOMm	0: Independent channel output
TAUAnTOCm	1: Set/reset mode
TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDEm	0: Disables dead time operation
TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDLm	
TAUAnTREm	0: Disables real-time output
TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRCm	
TAUAnTMEm	0: Disables modulation

Note The channel output mode can also be set to channel output mode controlled by software by setting TAUAnTOE.TAUAnTOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details, see *Table 15-12 “Channel output modes” on page 628*.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the one-pulse output function. Therefore, these registers must be set to 0.

Table 15-31 Simultaneous rewrite settings for one-pulse output function

Bit name	Setting
TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDMm	
TAUAnRDCm	

(5) Operating procedure for one-pulse output function

Table 15-32 Operating procedure for one-pulse output function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-28 "TAUAnCMORm settings for one-pulse output function" on page 668* and *Table 15-29 "TAUAnCMURm settings for one-pulse output function" on page 668*. Set the value of the TAUAnCDRm register. Set the channel output mode by setting the control bits as described in *Table 15-30 "Control bit settings for independent channel output mode 1" on page 669*.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TAUAnTEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a start edge is detected, TAUAnCNTm loads the TAUAnCDRm value.
During operation	The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at all times.	INTTAUAnIm is generated when TAUAnCNTm starts and TAUAnTTOUTm is set to its active level. TAUAnCNTm counts down. When the counter reaches 0001_H: • INTTAUAnIm is generated. • TAUAnTTOUTm is set to its inactive level. TAUAnCNTm stops counting and waits for a trigger. If a trigger occurs while TAUAnCNTm is counting, the trigger is ignored. Afterwards, this procedure is repeated. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

15.17 Independent Channel Signal Measurement Functions

This chapter describes functions that measure the widths of an individual TAUAnTTINm pulse or the total width of successive TAUAnTTINm pulses. It also describes functions that measure the interval of the signal or that compare the width of a pulse with a reference value.

- *15.17.1 “TAUAnTTINm input pulse interval measurement function”*
- *15.17.2 “TAUAnTTINm input signal width measurement function”*
- *15.17.3 “Overflow interrupt output function (during TAUAnTTINm width measurement)”*
- *15.17.4 “TAUAnTTINm input period count detection function”*
- *15.17.5 “Overflow interrupt output function (during TAUAnTTINm input period count detection)”*
- *15.17.6 “TAUAnTTINm input pulse interval judgment function”*
- *15.17.7 “TAUAnTTINm input signal width judgment function”*

15.17.1 TAUAnTTINm input pulse interval measurement function

(1) Overview

Summary This function captures the count value and uses this value and the overflow bit TAUAnCSRm.TAUAnOVF to measure the interval of the TAUAnTTINm input signal.

Prerequisites

- The operation mode must be set to capture mode. See *Table 15-34 "TAUAnCMORm settings for TAUAnTTINm input pulse interval measurement function" on page 674*.
- TAUAnTTOUTm is not used for this function.

Description The counter is started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. The counter TAUAnCNTm starts counting up from 0000_H. When a valid TAUAnTTINm edge is detected, the value of TAUAnCNTm is captured, transferred to TAUAnCDRm, and an interrupt INTTAUAnIm is generated. The counter resets to 0000_H and subsequently continues operation.

If the counter reaches $FFFF_H$ before a valid TAUAnTTINm edge is detected, it overflows. The counter is reset to 0000_H and subsequently continues operation. The values transferred to TAUAnCDRm and TAUAnCSRm.TAUAnOVF respectively depend on the values of bits TAUAnCMORm.TAUAnCOS[1:0]:

Table 15-33 Effects of an overflow

TAUAnCMORm. TAUAnCOS[1:0]	When overflow occurs		When a valid TAUAnTTINm input is then detected	
	TAUAnCDRm	TAUAnCSRm. TAUAnOVF	TAUAnCDRm and TAUAnCNTm	TAUAnCSRm. TAUAnOVF
00	Unchanged	0	TAUAnCNTm loaded to TAUAnCDRm	1
01		1		
10	Set to $FFFF_H$	0	TAUAnCNTm set to 0, TAUAnCDRm unchanged	0
11		1		

When TAUAnCMORm.TAUAnCOS[0] is 1, the overflow bit TAUAnCSRm.TAUAnOVF can only be cleared by setting TAUAnCSCm.TAUAnCLOV to 1.

The combination of the value of TAUAnCDRm and TAUAnCSRm.TAUAnOVF can be used to deduce the interval of the TAUAnTTINm signal. However, if an overflow occurs multiple times before a valid TAUAnTTINm input is detected, the overflow bit TAUAnCSRm.TAUAnOVF cannot indicate this.

The function can be stopped by setting TAUAnTT.TAUAnTTm = 1, which in turn sets TAUAnTE.TAUAnTEm = 0. TAUAnCNTm stops but retains its value. While the function is stopped, TAUAnTTINm input valid edge detection and TAUAnCNTm capture are not performed.

The function can be restarted by setting TAUAnTS.TAUAnTSm = 1. The counter is reset to 0000_H and subsequently continues operation. The counter can also be forcibly restarted (without stopping it first) by setting TAUAnTS.TAUAnTSm = 1 during operation.

Conditions If the TAUAnCMORm.TAUAnMD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details, see *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation When Counter Starts or Restarts" on page 639*.

Note When TAUAnCMORm.TAUAnCOS[1] = 1, the value of TAUAnCNTm is *not* loaded to TAUAnCDRm when the first valid TAUAnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUAnTTINm input pulse interval = count clock cycle x [(TAUAnCSRm.TAUAnOVF x ($FFFF_H$ + 1)) + TAUAnCDRm capture value + 1]

(3) Block diagram and general timing diagram

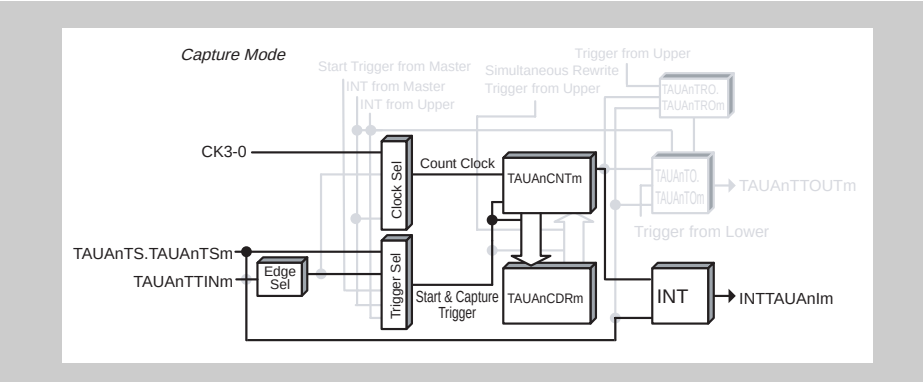


Figure 15-42 Block diagram for TAUAnTTINm input pulse interval measurement function

The following settings apply to the general timing diagram:

- INTTAUAnIm not generated at operation start (TAUAnCMORm.TAUAnMD0 = 0)
- Falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)
- When a valid TAUAnTTINm input is detected after an overflow TAUAnCDRm is changed and TAUAnCSRm.TAUAnOVF is set to 1 (TAUAnCMORm.TAUAnCOS[1:0] = 00_B)

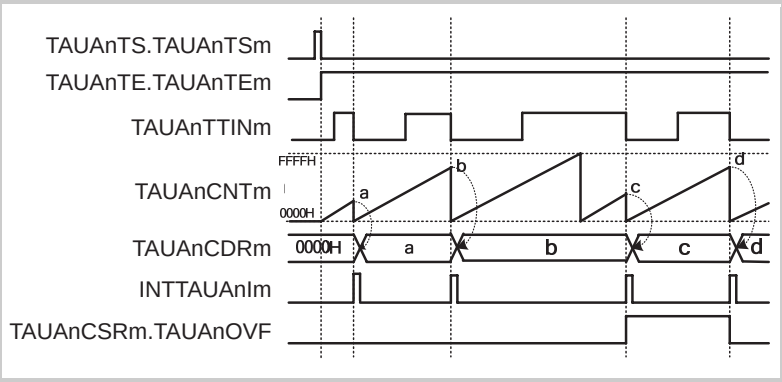


Figure 15-43 General timing diagram for TAUAnTTINm input pulse interval measurement function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-34 TAUAnCMORm settings for TAUAnTTINm input pulse interval measurement function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	001: Valid edge of the TAUAnTTINm input signal is the external capture trigger
TAUAnCOS[1:0]	See *Table 15-33 "Effects of an overflow" on page 672*.
TAUAnMD[4:1]	0010: Capture mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start 1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-35 TAUAnCMURm settings for TAUAnTTINm input pulse interval measurement function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, the channel output mode can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm input pulse interval measurement function. Therefore, these registers must be set to 0.

Table 15-36 Simultaneous rewrite settings for TAUAnTTINm input pulse interval measurement function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(5) Operating procedure for TAUAnTTINm input pulse interval measurement function

Table 15-37 Operating procedure for TAUAnTTINm input pulse interval measurement function

		Operation	Status of TAUAn
	Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-34 "TAUAnCMORm settings for TAUAnTTINm input pulse interval measurement function" on page 674* and *Table 15-35 "TAUAnCMURm settings for TAUAnTTINm input pulse interval measurement function" on page 674*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Restart	Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm is cleared to 0000_H. INTTAUAnIm is generated when TAUAnCMORm.TAUAnMD0 is set to 1.
	During operation	Detection of TAUAnTTINm edges. The TAUAnCMURm.TAUAnTIS[1:0] bits can be changed at any time. The TAUAnCDRm and TAUAnCSRm registers can be read at any time. Writing 1 to the TAUAnCSCm.TAUAnCLOV bit is possible. (The TAUAnCSRm.TAUAnOVF bit is cleared to 0.)	• TAUAnCNTm transfers (captures) its value to TAUAnCDRm and retains its value. INTTAUAnIm is then generated. Counting stops at "value transferred to TAUAnCDRm + 1" and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. This procedure is then repeated.
	Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnCSRm.TAUAnOVF retain their current values.

(6) Specific timing diagrams: overflow behavior

(a) TAUAnCMORm.TAUAnCOS[1:0] = 00_B

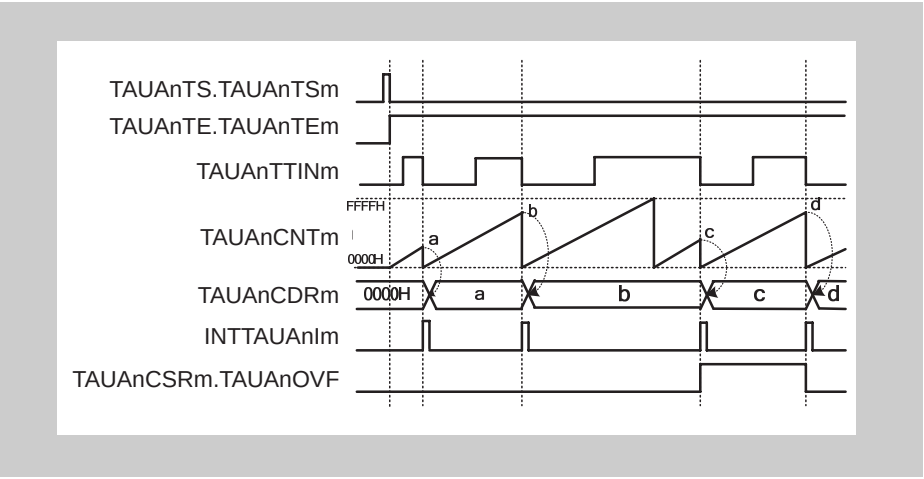


Figure 15-44 TAUAnCMORm.TAUAnCOS[1:0] = 00_B, TAUAnCMORm.TAUAnMD0 = 0, TAUAnCMURm.TAUAnTIS[1:0] = 00_B

- When an overflow occurs, the value of TAUAnCDRm remains unchanged and TAUAnCSRm.TAUAnOVF remains = 0.
- Upon detection of the next valid TAUAnTTINm input edge, the value of TAUAnCNTm is loaded to TAUAnCDRm and TAUAnCSRm.TAUAnOVF is set to 1.
- Upon detecting the next valid TAUAnTTINm input edge while no overflow has occurred, TAUAnCSRm.TAUAnOVF is cleared to 0.

(b) TAUAnCMORm.TAUAnCOS[1:0] = 01_B

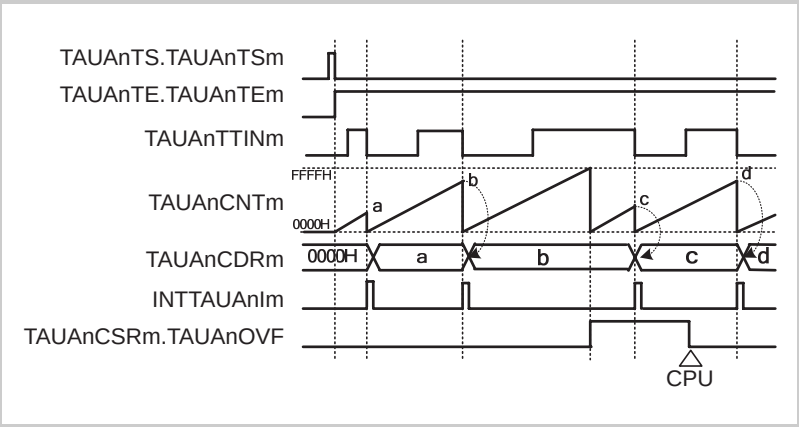


Figure 15-45 TAUAnCMORm.TAUAnCOS[1:0] = 01_B, TAUAnCMORm.TAUAnMD0 = 0, TAUAnCMURm.TAUAnTIS[1:0] = 00_B

- When an overflow occurs, the value of TAUAnCDRm remains unchanged and TAUAnCSRm.TAUAnOVF is set to 1.
- Upon detection of the next valid TAUAnTTINm input edge, the value of TAUAnCNTm is loaded to TAUAnCDRm.
- TAUAnCSRm.TAUAnOVF is only cleared by a CPU command (setting the TAUAnCSCm.TAUAnCLOV bit to 1).

(c) TAUAnCMORm.TAUAnCOS[1:0] = 10_B

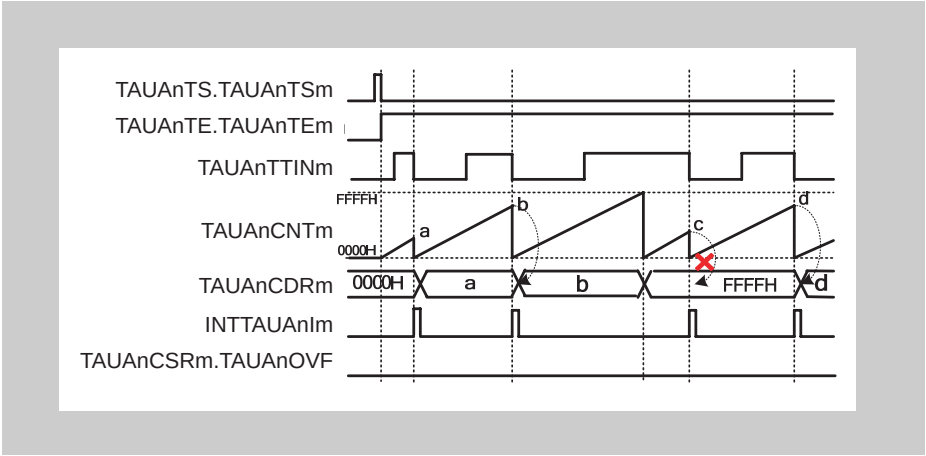


Figure 15-46 TAUAnCMORm.TAUAnCOS[1:0] = 10_B, TAUAnCMORm.TAUAnMD0 = 0, TAUAnCMURm.TAUAnTIS[1:0] = 00_B

- When an overflow occurs, TAUAnCDRm is set to $FFFF_H$ and TAUAnCSRm.TAUAnOVF remains = 0.
- Upon detection of the next valid TAUAnTTINm input edge, TAUAnCNTm is reset to 0, but TAUAnCDRm and TAUAnCSRm.TAUAnOVF remain unchanged.
- Thus, the next TAUAnTTINm input valid edge after the overflow is ignored.

(d) TAUAnCMORm.TAUAnCOS[1:0] = 11_B

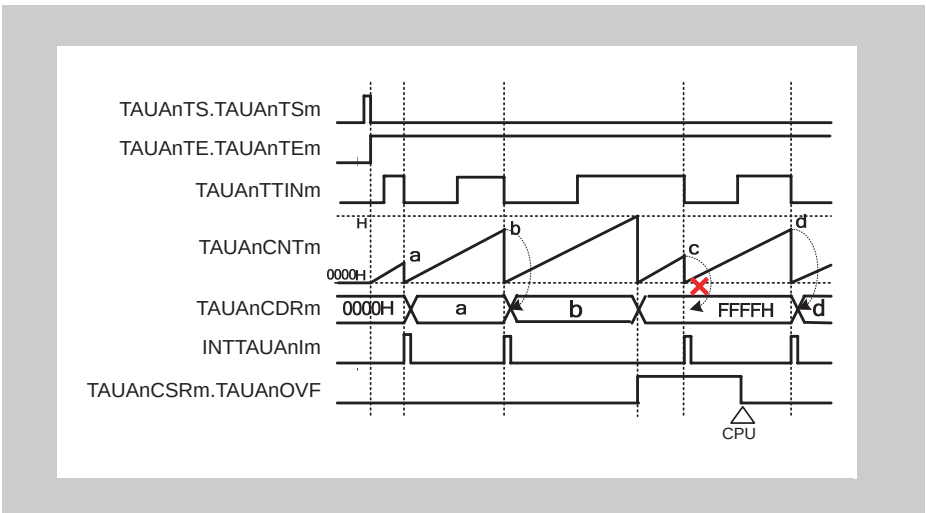


Figure 15-47 TAUAnCMORm.TAUAnCOS[1:0] = 11_B, TAUAnCMORm.TAUAnMD0 = 0, TAUAnCMURm.TAUAnTIS[1:0] = 00_B

- When an overflow occurs, TAUAnCDRm is set to $FFFF_H$, and TAUAnCSRm.TAUAnOVF is set to 1.
- Upon detection of the next valid TAUAnTTINm input edge, TAUAnCNTm is reset to 0, but TAUAnCDRm and TAUAnCSRm.TAUAnOVF remain unchanged.
- Thus, the next TAUAnTTINm input valid edge after the overflow is ignored.
- TAUAnCSRm.TAUAnOVF is cleared by setting TAUAnCSCm.TAUAnCLOV = 1.

15.17.2 TAUAnTTINm input signal width measurement function

(1) Overview

Summary This function measures the width of a TAUAnTTINm input signal.

Prerequisites

- The operation mode must be set to capture & one-count mode. See *Table 15-39 "TAUAnCMORm settings for TAUAnTTINm input signal width measurement function" on page 681*.
- TAUAnTTOUTm is not used for this function.
- TAUAnCMORm.TAUAnMD0 must be set to 0.

Description The counter is started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. When a valid TAUAnTTINm start edge is detected, the counter TAUAnCNTm starts counting up from 0000_H. When a valid TAUAnTTINm stop edge is detected, the value of TAUAnCNTm is captured, transferred to TAUAnCDRm, and an interrupt INTTAUAnIm is generated. The counter retains its value <R> (TAUAnCDRm + 1) and awaits the next valid TAUAnTTINm input start edge.

If the counter reaches $FFFF_H$ before a valid TAUAnTTINm stop edge is detected, it overflows. The counter is reset to 0000_H and subsequently continues operation. The values transferred to TAUAnCDRm and TAUAnCSRm.TAUAnOVF respectively depend on the values of bits TAUAnCMORm.TAUAnCOS[1:0].

Table 15-38 Effects of an overflow

TAUAnCMORm. COS[1:0]	When overflow occurs		When a valid TAUAnTTINm input stop edge is detected	
	TAUAnCDRm	TAUAnCSRm. TAUAnOVF	TAUAnCDRm and TAUAnCNTm	TAUAnCSRm. TAUAnOVF
00	Unchanged	0	TAUAnCNTm loaded to TAUAnCDRm	1
01		1		
10	Set to $FFFF_H$	0	TAUAnCNTm stops counting TAUAnCDRm unchanged	0
11		1		

When TAUAnCMORm.TAUAnCOS[0] is 1, the overflow bit TAUAnCSRm.TAUAnOVF can only be cleared by setting TAUAnCSCm.TAUAnCLOV to 1.

The combination of the value of TAUAnCDRm and TAUAnCSRm.TAUAnOVF can be used to deduce the width of the TAUAnTTINm signal. However, if an overflow occurs multiple times before a valid TAUAnTTINm input is detected, the overflow bit TAUAnCSRm.TAUAnOVF cannot indicate this.

This function cannot be forcibly restarted.

Note When TAUAnCMORm.TAUAnCOS[2] = 1_B, the value of TAUAnCNTm is *not* loaded to TAUAnCDRm when the first valid TAUAnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUAnTTINm input signal width = count clock cycle x [(TAUAnCSRm.TAUAnOVF x ($FFFF_H$ + 1)) + TAUAnCDRm capture value + 1]

(3) Block diagram and general timing diagram

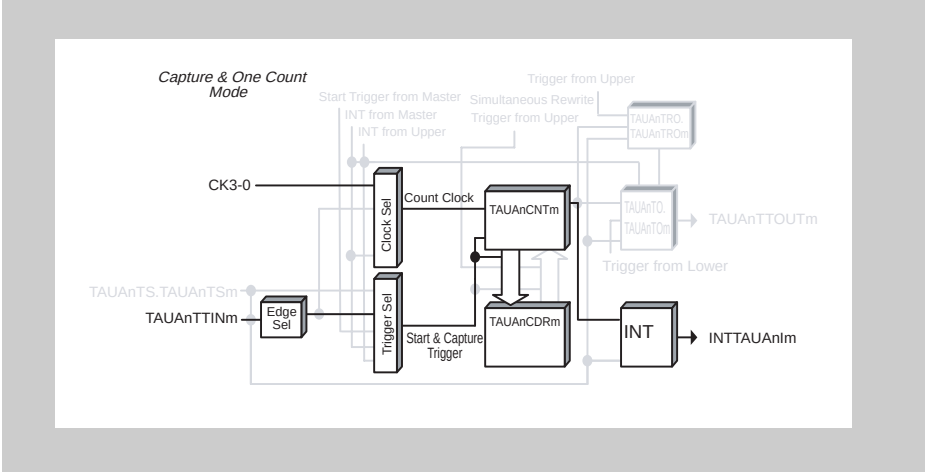


Figure 15-48 Block diagram for TAUAnTTINm input signal width measurement function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TAUAnTIS[1:0] = 11_B)
- When a valid TAUAnTTINm input is detected after an overflow TAUAnCDRm is changed and TAUAnCSRm.TAUAnOVF is set to 1 (TAUAnCMORm.TAUAnCOS[1:0] = 00_B)

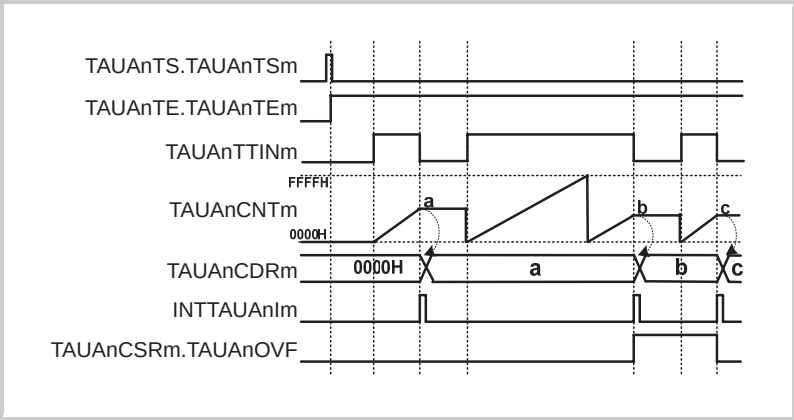


Figure 15-49 General timing diagram for TAUAnTTINm input signal width measurement function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-39 TAUAnCMORm settings for TAUAnTTINm input signal width measurement function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
TAUAnCOS[1:0]	See *Table 15-38 “Effects of an overflow” on page 679*.
TAUAnMD[4:1]	0110: Capture & one-count mode
TAUAnMD0	0: Disables the start trigger during operation

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-40 TAUAnCMURm settings for TAUAnTTINm input signal width measurement function

Bit name	Setting
TAUAnTIS[1:0]	10:Rising and falling edge detection (low width measurement) 11:Rising and falling edge detection (high width measurement)

(c) Channel output mode

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm input signal width measurement function. Therefore, these registers must be set to 0.

Table 15-41 Simultaneous rewrite settings for TAUAnTTINm input signal width measurement function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(5) Operating procedure for TAUAnTTINm input signal width measurement function

Table 15-42 Operating procedure for TAUAnTTINm input signal width measurement function

<R>

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-39 "TAUAnCMORm settings for TAUAnTTINm input signal width measurement function" on page 681* and *Table 15-40 "TAUAnCMURm settings for TAUAnTTINm input signal width measurement function" on page 681*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a TAUAnTTINm start is detected, TAUAnCNTm starts to count up.
During operation	Detection of TAUAnTTINm edges. The TAUAnCDRm, TAUAnCNTm, and TAUAnCSRm registers can be read at any time. The TAUAnCSC.CLOV bit can be set to 1.	TAUAnCNTm starts to count up from 0000_H. When a TAUAnTTINm valid edge is detected: • TAUAnCNTm transfers (captures) its value to TAUAnCDRm, and INTTAUAnIm is then generated. • Counting stops at the "value transferred to TAUAnCDRm + 1" value and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnCSRm.TAUAnOVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUAnCMORm.TAUAnCOS[1:0] = 00_B

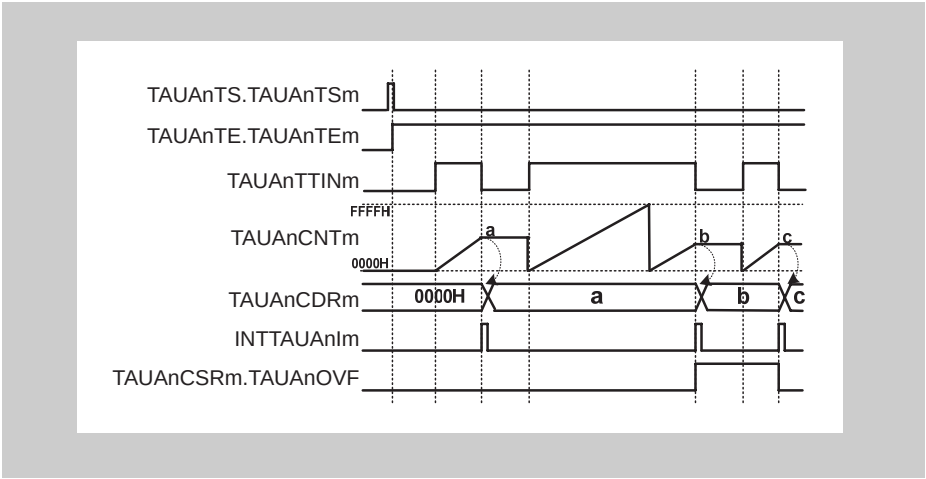


Figure 15-50 TAUAnCMORm.TAUAnCOS[1:0] = 00_B, TAUAnCMORm.TAUAnMD0 = 0, TAUAnCMURm.TAUAnTIS[1:0] = 11_B

- When an overflow occurs, the value of TAUAnCDRm remains unchanged and TAUAnCSRm.TAUAnOVF remains = 0.
- Upon detection of the next valid TAUAnTTINm input edge, the value of TAUAnCNTm is loaded to TAUAnCDRm and TAUAnCSRm.TAUAnOVF is set to 1.
- Upon detecting the next valid TAUAnTTINm input edge while no overflow has occurred, TAUAnCSRm.TAUAnOVF is cleared to 0.

(b) TAUAnCMORm.TAUAnCOS[1:0] = 01_B

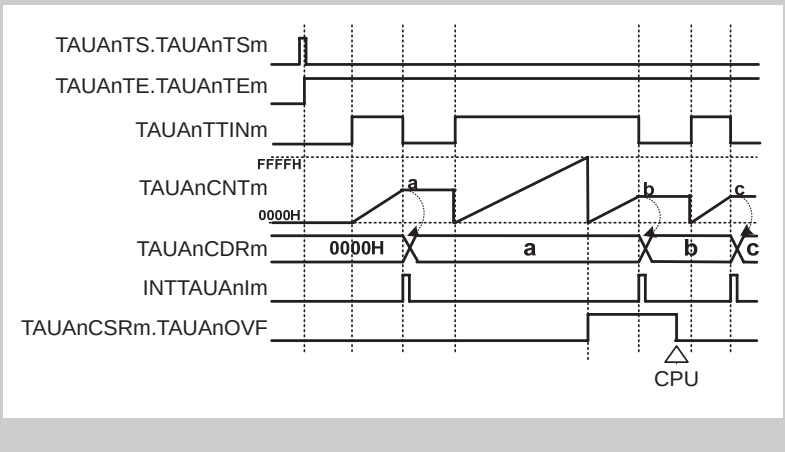


Figure 15-51 TAUAnCMORm.TAUAnCOS[1:0] = 01_B, TAUAnCMORm.TAUAnMD0 = 0, TAUAnCMURm.TAUAnTIS[1:0] = 11_B

- When an overflow occurs, the value of TAUAnCDRm remains unchanged and TAUAnCSRm.TAUAnOVF is set to 1.
- Upon detection of the next valid TAUAnTTINm input edge, the value of TAUAnCNTm is loaded to TAUAnCDRm.
- TAUAnCSRm.TAUAnOVF is only cleared by a CPU command (setting TAUAnCSCm.TAUAnCLOV = 1).

(c) TAUAnCMORm.TAUAnCOS[1:0] = 10_B

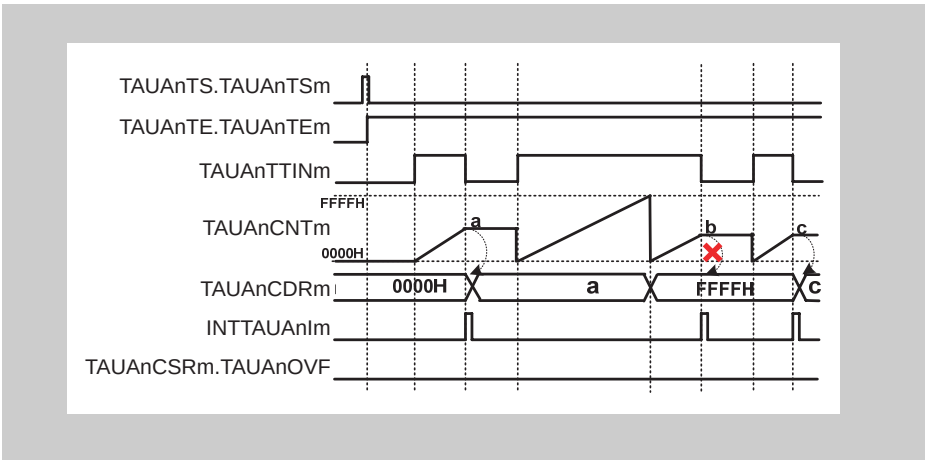


Figure 15-52 TAUAnCMORm.TAUAnCOS[1:0] = 10_B, TAUAnCMORm.TAUAnMD0 = 0, TAUAnCMURm.TAUAnTIS[1:0] = 11_B

- When an overflow occurs, TAUAnCDRm is set to $FFFF_H$ and TAUAnCSRm.TAUAnOVF remains = 0.
- Upon detection of the next valid TAUAnTTINm input edge, TAUAnCNTm is reset to 0, but TAUAnCDRm and TAUAnCSRm.TAUAnOVFTAUAnCSRm.TAUAnOVF remain unchanged.
- Thus, the next TAUAnTTINm input valid edge after the overflow is ignored.

(d) TAUAnCMORm.TAUAnCOS[1:0] = 11_B

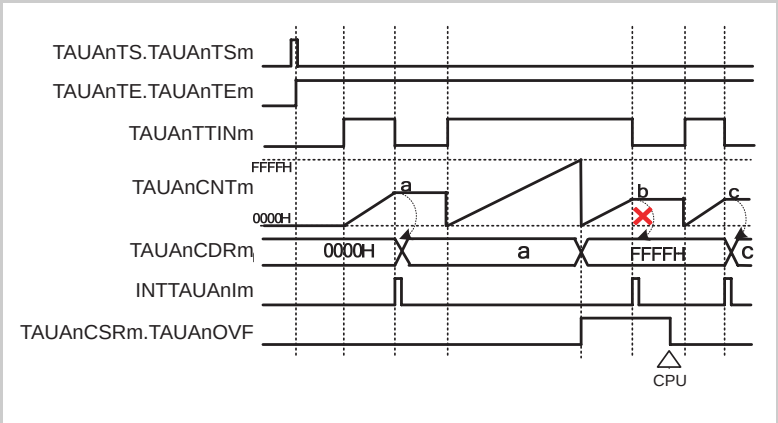


Figure 15-53 TAUAnCMORm.TAUAnCOS[1:0] = 11_B, TAUAnCMORm.TAUAnMD0 = 0, TAUAnCMURm.TAUAnTIS[1:0] = 11_B

- When an overflow occurs, TAUAnCDRm is set to $FFFF_H$, and TAUAnCSRm.TAUAnOVF is set to 1.
- Upon detection of the next valid TAUAnTTINm input edge, TAUAnCNTm is reset to 0, but TAUAnCDRm and TAUAnCSRm.TAUAnOVF remain unchanged.
- Thus, the next TAUAnTTINm input valid edge after the overflow is ignored.
- TAUAnCSRm.TAUAnOVF is cleared by setting TAUAnCSCm.TAUAnCLOV = 1.

15.17.3 Overflow interrupt output function (during TAUAnTTINm width measurement)

(1) Overview

Summary This function measures the width of an individual TAUAnTTINm input signal. An interrupt is generated if (FFFFH + 1) is exceeded following TAUAnTTINm input.

Prerequisites
- The operation mode must be set to one-count mode. See *Table 15-43 "TAUAnCMORm settings for overflow interrupt output function (during TAUAnTTINm width measurement)" on page 688*.
- TAUAnTTOUTm is not used for this function.
- The value of TAUAnCDRm must be set to $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input start edge is detected. $FFFF_H$ is loaded to TAUAnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value.

When the next TAUAnTTINm input start edge is detected, TAUAnCNTm loads $FFFF_H$ and starts to count down.

If the counter reaches 0000_H before a stop edge is detected, an interrupt is generated.

Conditions The valid start and stop edges are specified by the TAUAnCMURm.TAUAnTIS[1:0] bits:
- If TAUAnCMURm.TAUAnTIS[1:0] = 10_B, the TAUAnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUAnCMURm.TAUAnTIS[1:0] = 11_B, the TAUAnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

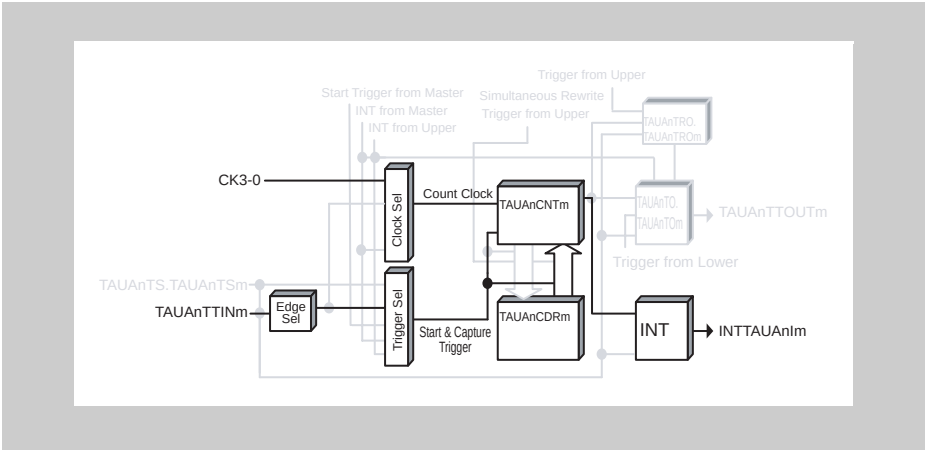


Figure 15-54 Block diagram for overflow interrupt output function (during TAUAnTTINm width measurement)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TAUAnTIS[1:0] = 11_B)

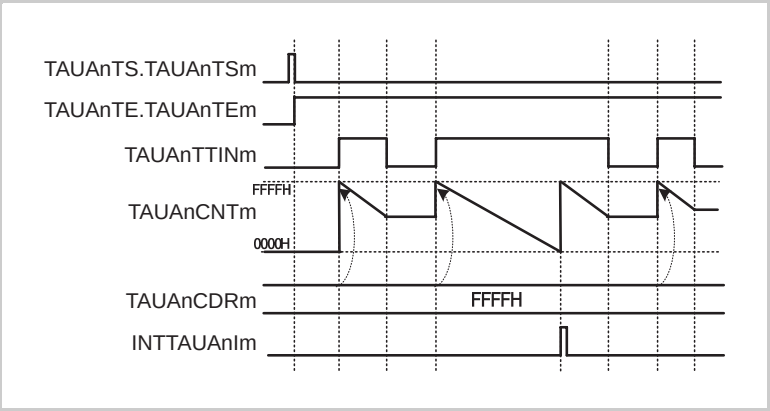


Figure 15-55 General timing diagram for overflow interrupt output function (during TAUAnTTINm width measurement)

(3) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-43 TAUAnCMORm settings for overflow interrupt output function (during TAUAnTTINm width measurement)

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	0: Disables the start trigger during operation

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-44 TAUAnCMURm settings overflow interrupt output function (during TAUAnTTINm width measurement)

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

Because channel output mode is not used by this function, TAUAnTOE.TAUAnTOEm is cleared. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the overflow interrupt output function (during TAUAnTTINm width measurement). Therefore, these registers must be set to 0.

(4) Operating procedure for overflow interrupt output function

Table 15-45 Simultaneous rewrite settings for overflow interrupt output function (during TAUAnTTINm width measurement)

Bit name	Setting
TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDMm	
TAUAnRDCm	

(during TAUAnTTINm width measurement)

Table 15-46 Operating procedure for overflow interrupt output function (during TAUAnTTINm width measurement)

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-43 "TAUAnCMORm settings for overflow interrupt output function (during TAUAnTTINm width measurement)" on page 688* and *Table 15-44 "TAUAnCMURm settings overflow interrupt output function (during TAUAnTTINm width measurement)" on page 688*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TAUAnTEm is set to 1 and TAUAnCNTm waits for detection of the start edge. When a start edge is detected, TAUAnCNTm loads the TAUAnCDRm value ($FFFF_H$).
During operation	The TAUAnCNTm register can be read at any time. Detection of TAUAnTTINm edges.	TAUAnCNTm counts down. When the counter reaches 0000_H: • INTTAUAnIm is generated. When a reverse edge of TAUAnTTINm is detected during count operation: • TAUAnCNTm stops counting and waits for a trigger. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart →

15.17.4 TAUAnTTINm input period count detection function

(1) Overview

Summary This function measures the cumulative width of a TAUAnTTINm input signal.

Prerequisites
- The operation mode must be set to capture & gate count mode. See *Table 15-47 "TAUAnCMORm settings for TAUAnTTINm input period count detection function" on page 693*.
- TAUAnTTOUTm is not used for this function.

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. The counter awaits a valid TAUAnTTINm input edge.

When a valid TAUAnTTINm input start edge is detected, the counter starts to count from 0000_H.

<R> When a valid TAUAnTTINm input stop edge is detected, the current TAUAnCNTm value is loaded to TAUAnCDRm and an interrupt (INTTAUAnIm) is generated. The counter stops and retains its value (TAUAnCDRm + 1) until the next valid TAUAnTTINm input start edge is detected.

When the next valid TAUAnTTINm input start edge is detected, the counter resumes counting, starting with its value upon stopping.

If the counter reaches $FFFF_H$ the bit TAUAnCSRm.TAUAnOVF is set to 1 and the counter restarts from 0000_H. The value of TAUAnCSRm.TAUAnOVF is reset by the CPU by setting TAUAnCSCm.TAUAnCLOV = 1.

Note The input TAUAnTTINm is sampled at the frequency of the operation clock, specified by the TAUAnCMORm.TAUAnCKS[1:0] bits.

Conditions The valid start and stop edges are specified by the TAUAnCMURm.TAUAnTIS[1:0] bits:
- If TAUAnCMURm.TAUAnTIS[1:0] = 10_B, the TAUAnTTINm input low period is counted. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUAnCMURm.TAUAnTIS[1:0] = 11_B, the TAUAnTTINm input high period is counted. The start trigger is a rising edge and the stop trigger is a falling edge.

(2) Equations

<R> Cumulative TAUAnTTINm input width =
count clock cycle × [(($FFFF_H$ + 1) × TAUAnCSRm.TAUAnOVF) + (TAUAnCDRm capture value + 1)]

(3) Block diagram and general timing diagram

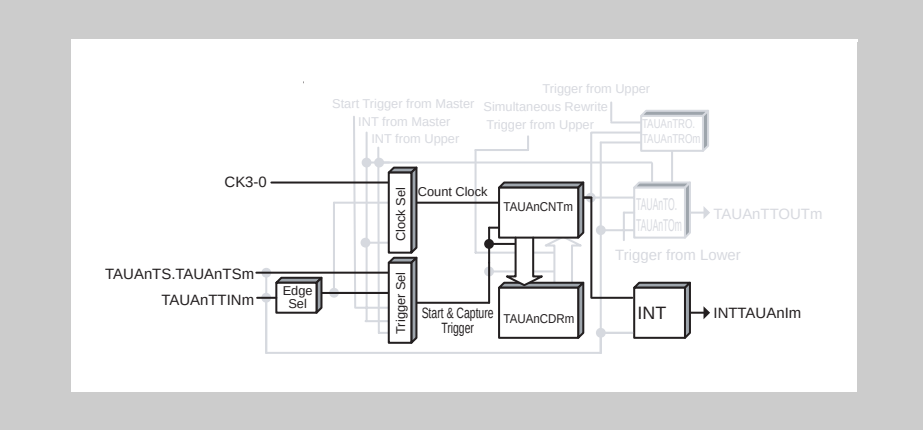


Figure 15-56 Block diagram for TAUAnTTINm input period count detection function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TAUAnTIS[1:0] = 11_B)

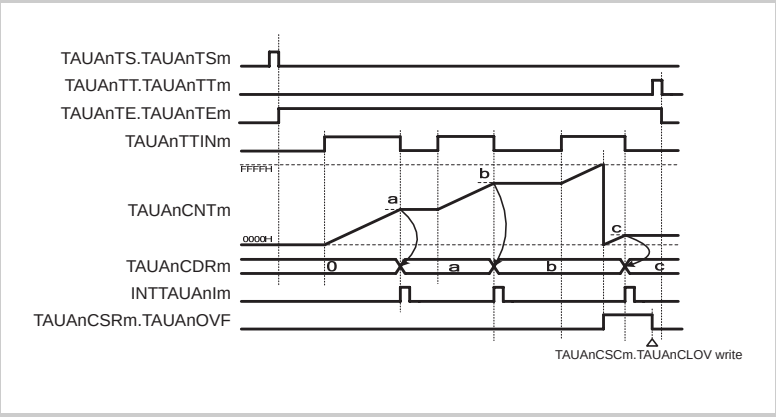


Figure 15-57 General timing diagram for TAUAnTTINm input period count detection function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-47 TAUAnCMORm settings for TAUAnTTINm input period count detection function

<R>

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
TAUAnCOS[1:0]	01: Overflow (TAUAnCSRm.TAUAnOVF) set upon counter overflow and cleared by a CPU instruction (by setting the TAUAnCSCm.TAUAnCLOV bit to 1).
TAUAnMD[4:1]	1101: Capture & gate count mode
TAUAnMD0	0: Disables the start trigger during operation

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-48 TAUAnCMURm settings for TAUAnTTINm input period count detection function

Bit name	Setting
TAUAnTIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm input period count detection function. Therefore, these registers must be set to 0.

Table 15-49 Simultaneous rewrite settings for TAUAnTTINm input period count detection function

Bit name	Setting
TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDMm	
TAUAnRDCm	

(5) Operating procedure for TAUAnTTINm input period count detection function

Table 15-50 Operating procedure for TAUAnTTINm Input Period Count Detection Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-47 "TAUAnCMORm settings for TAUAnTTINm input period count detection function" on page 693* and *Table 15-48 "TAUAnCMURm settings for TAUAnTTINm input period count detection function" on page 693*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TAUAnTEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a start edge is detected, TAUAnCNTm is cleared to 0000_H and TAUAnCNTm starts to count up.
During operation	Detection of TAUAnTTINm edges. The TAUAnCDRm, TAUAnCNTm, and TAUAnCSRm registers can be read at any time. The TAUAnCSCm.TAUAnCLOV bit can be set to 1.	When a TAUAnTTINm start edge (rising edge for high width measurement, falling edge for low width measurement) is detected, TAUAnCNTm starts to count up from the stop value. When TAUAnCNTm detects a stop edge (falling edge for high width measurement, rising edge for low width measurement), it transfers the value to TAUAnCDRm and INTTAUAnIm is generated. Counting stops at the "value transferred to TAUAnCDRm + 1" value and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. If the TAUAnCNTm reaches $FFFF_H$, the counter overflows and TAUAnCSRm.TAUAnOVF is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and it and TAUAnCSRm.TAUAnOVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

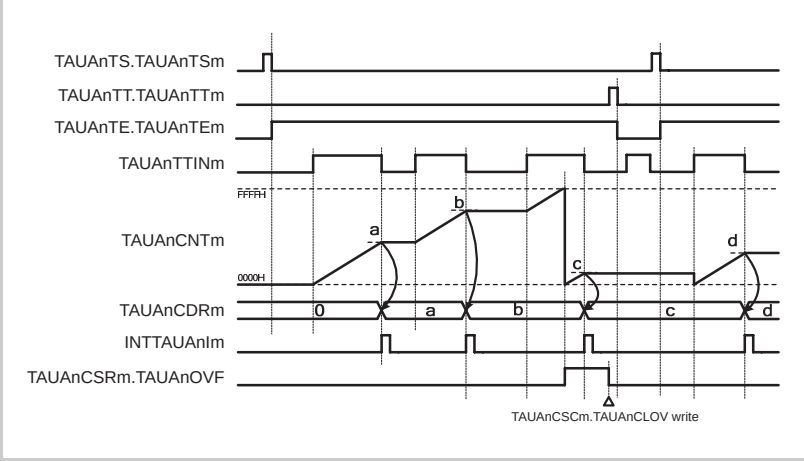


Figure 15-58 Operation stop and restart, TAUAnCMURm.TAUAnTIS[1:0] = 11_B

- The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1, which in turn sets TAUAnTE.TAUAnTEm to 0.
- TAUAnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUAnTTINm input edges are ignored.
- The counter can be restarted by setting TAUAnTS.TAUAnTSm to 1. TAUAnCNTm restarts to count from 0000_H.

15.17.5 Overflow interrupt output function (during TAUAnTTINm input period count detection)

(1) Overview

Summary This function measures the cumulative width of a TAUAnTTINm input signal. If the cumulative TAUAnTTINm input width is longer than $FFFF_H$, an interrupt can be generated to output an overflow interrupt.

Prerequisites
- The operation mode must be set to gate count mode. See *Table 15-51 "TAUAnCMORm settings for overflow interrupt output function (during TAUAnTTINm input period count detection)" on page 699*.
- TAUAnTTOUTm is not used for this function.
- The value of TAUAnCDRm must be set to $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input start edge is detected. $FFFF_H$ is loaded to TAUAnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value. The counter awaits the next TAUAnTTINm input start edge and then continues to count down from the current value.

When the counter reaches 0000_H an interrupt is generated. $FFFF_H$ is loaded to TAUAnCNTm and the counter continues to count down until a TAUAnTTINm input stop edge is detected.

Conditions The valid start and stop edges are specified by the TAUAnCMURm.TAUAnTIS[1:0] bits:
- If TAUAnCMURm.TAUAnTIS[1:0] = 10_B, the TAUAnTTINm input low period is counted. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUAnCMURm.TAUAnTIS[1:0] = 11_B, the TAUAnTTINm input high period is counted. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

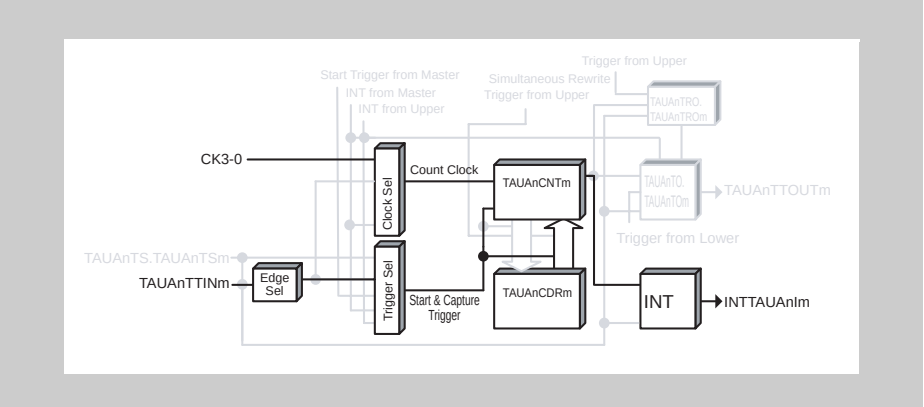


Figure 15-59 Block diagram for overflow interrupt output function (during TAUAnTTINm input period count detection)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TAUAnTIS[1:0] = 11_B)

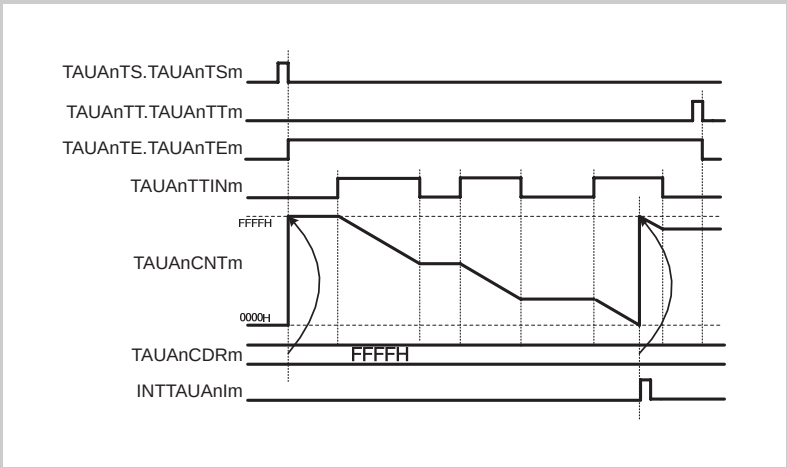


Figure 15-60 General timing diagram for overflow interrupt output function (during TAUAnTTINm input period count detection)

(3) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-51 TAUAnCMORm settings for overflow interrupt output function (during TAUAnTTINm input period count detection)

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	1100: Gate count mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-52 TAUAnCMURm settings for overflow interrupt output function (during TAUAnTTINm input period count detection)

Bit name	Setting
TAUAnTIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the overflow interrupt output function (during TAUAnTTINm input period count detection). Therefore, these registers must be set to 0.

Table 15-53 Simultaneous rewrite settings for overflow interrupt output function (during TAUAnTTINm input period count detection)

Bit name	Setting
TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDMm	
TAUAnRDCm	

(4) Operating procedure for overflow interrupt output function (during TAUAnTTINm input period count detection)

Table 15-54 Operating procedure for overflow interrupt output function (during TAUAnTTINm input period count detection)

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-51 "TAUAnCMORm settings for overflow interrupt output function (during TAUAnTTINm input period count detection)" on page 699* and *Table 15-52 "TAUAnCMURm settings for overflow interrupt output function (during TAUAnTTINm input period count detection)" on page 699*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TAUAnTEm is set to 1 and TAUAnCNTm waits for detection of the start edge. When a start edge is detected, TAUAnCNTm loads the TAUAnCDRm value ($FFFF_H$).
During operation	The TAUAnCNTm register can be read at all times.	TAUAnCNTm counts down. When the counter reaches 0000_H: • INTTAUAnIm is generated. • TAUAnCNTm loads the TAUAnCDRm value ($FFFF_H$), and then continues counting down. When a reverse edge of TAUAnTTINm is detected during count operation: • TAUAnCNTm counts down from the stop value. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart

15.17.6 TAUAnTTINm input pulse interval judgment function

(1) Overview

Summary This function outputs the result of a comparison between the count value (TAUAnCNTm) and the value in the channel data register (TAUAnCDRm) when a TAUAnTTINm input pulse occurs. An interrupt signal INTTAUAnIm is generated if the result of the comparison is true.

Prerequisites
- The operation mode must be set to judge mode. See *Table 15-55 "TAUAnCMORm settings for TAUAnTTINm input pulse interval judgment function" on page 704*.
- TAUAnTTOUTm is not used for this function.

Description The counter is started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. The current value of TAUAnCDRm is loaded to TAUAnCNTm and the counter starts to count down from this value.

When a TAUAnTTINm valid edge is detected or TAUAnTS.TAUAnTSm is set to 1, the function compares the current values of TAUAnCNTm and TAUAnCDRm. An interrupt signal INTTAUAnIm is generated if the result of the comparison is true. TAUAnCNTm reloads the value of TAUAnCDRm, and then subsequently continues operation, regardless of the result of the comparison.

If the counter reaches 0000_H before a TAUAnTTINm valid edge is detected, TAUAnCNTm overflows and is set to $FFFF_H$. It then continues to count down.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the function starts to count down.

Conditions The TAUAnCMORm.TAUAnMD0 bit specifies the type of comparison:
- If TAUAnCMORm.TAUAnMD0 = 0, INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm.
- If TAUAnCMORm.TAUAnMD0 = 1, INTTAUAnIm is generated when TAUAnCNTm > TAUAnCDRm.

(2) Block diagram and general timing diagram

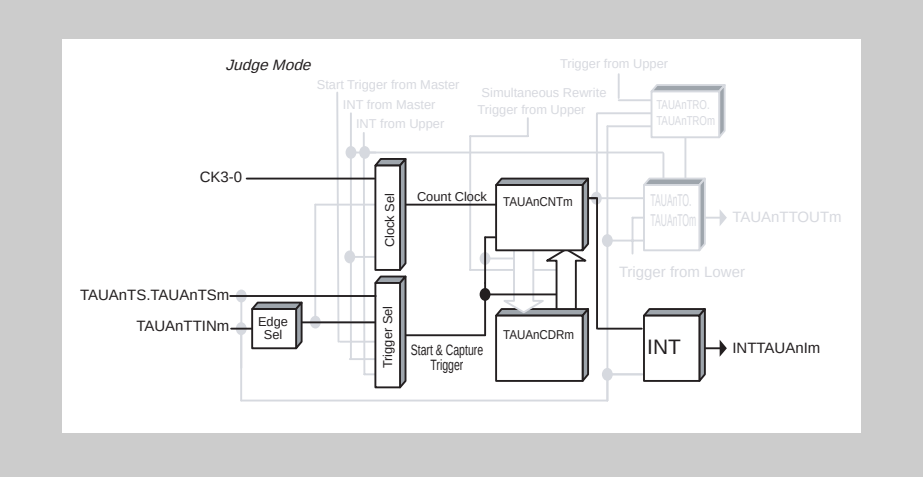


Figure 15-61 Block diagram for TAUAnTTINm input pulse interval judgment function

The following settings apply to the general timing diagram:

- Falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)

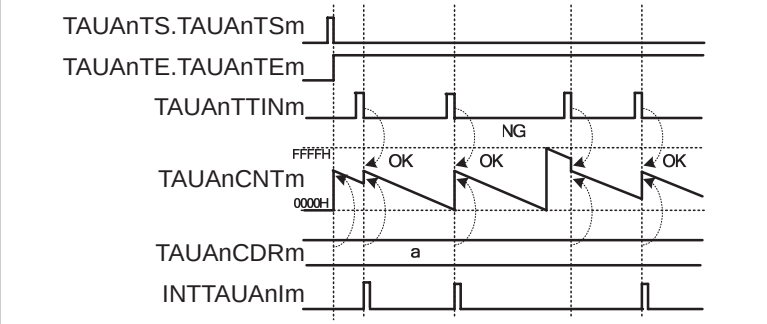


Figure 15-62 General timing diagram for TAUAnTTINm input pulse interval judgment function

(3) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-55 TAUAnCMORm settings for TAUAnTTINm input pulse interval judgment function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	001: Valid edge of the TAUAnTTINm input signal is the external start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0001: Judge mode
TAUAnMD0	0: INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm 1: INTTAUAnIm is generated when TAUAnCNTm > TAUAnCDRm

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-56 TAUAnCMURm settings for TAUAnTTINm input pulse interval judgment function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

<R>
<R>

(c) Channel output mode

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm input pulse interval judgment function. Therefore, these registers must be set to 0.

Table 15-57 Simultaneous rewrite settings for TAUAnTTINm input pulse interval judgment function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(4) Operating procedure for TAUAnTTINm input pulse interval judgment function

Table 15-58 Operating procedure for TAUAnTTINm input pulse interval judgment function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-55 "TAUAnCMORm settings for TAUAnTTINm input pulse interval judgment function" on page 704* and *Table 15-56 "TAUAnCMURm settings for TAUAnTTINm input pulse interval judgment function" on page 704*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value.
During operation	Detection of TAUAnTTINm edges. The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at any time.	TAUAnCNTm counts down. When a TAUAnTTINm input edge is detected • TAUAnCNTm reloads TAUAnCDRm, and then continues count operation. • TAUAnCNTm compares the values and judges the condition according to the TAUAnCMORm.TAUAnMD0 setting. • If the condition is satisfied, INTTAUAnIm is generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart

15.17.7 TAUAnTTINm input signal width judgment function

(1) Overview

Summary This function outputs the result of a comparison between the count value (TAUAnCNTm) and the value in the channel data register (TAUAnCDRm) at a valid stop edge of a TAUAnTTINm input signal. An interrupt signal INTTAUAnIm is generated if the result of the comparison is true.

Prerequisites
- The operation mode must be set to judge & one-count mode. See *Table 15-59 "TAUAnCMORm settings for TAUAnTTINm input signal width judgment function" on page 709*.
- TAUAnTTOUTm is not used for this function.

Description The counter is started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. When a valid TAUAnTTINm input start edge is detected, the current value of TAUAnCDRm is loaded to TAUAnCNTm and the counter starts to count down from this value.

When a TAUAnTTINm valid stop edge is detected, the function compares the current values of TAUAnCNTm and TAUAnCDRm. An interrupt signal INTTAUAnIm is generated if the result of the comparison is true. The counter TAUAnCNTm retains its value until the next TAUAnTTINm valid start edge is detected, regardless of the result of the comparison.

If the counter reaches 0000_H before a valid TAUAnTTINm stop edge is detected, TAUAnCNTm overflows and is set to $FFFF_H$. It then continues to count down.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the function starts to count down.

Conditions
- The TAUAnCMORm.TAUAnMD0 bit specifies the type of comparison:
  - If TAUAnCMORm.TAUAnMD0 = 0, INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm.
  - If TAUAnCMORm.TAUAnMD0 = 1, INTTAUAnIm is generated when TAUAnCNTm > TAUAnCDRm.
- The TAUAnCMURm.TAUAnTIS[1:0] bits specify the type of width measurement:
  - For high width measurement (TAUAnCMURm.TAUAnTIS[1:0] = 11_B), the start edge is a rising TAUAnTTINm edge and the stop edge is a falling TAUAnTTINm edge.
  - For low width measurement (TAUAnCMURm.TAUAnTIS[1:0] = 10_B), the start edge is a falling TAUAnTTINm edge and the stop edge is a rising TAUAnTTINm edge.
- Forced restart is not possible for this function.

(2) Block diagram and general timing diagram

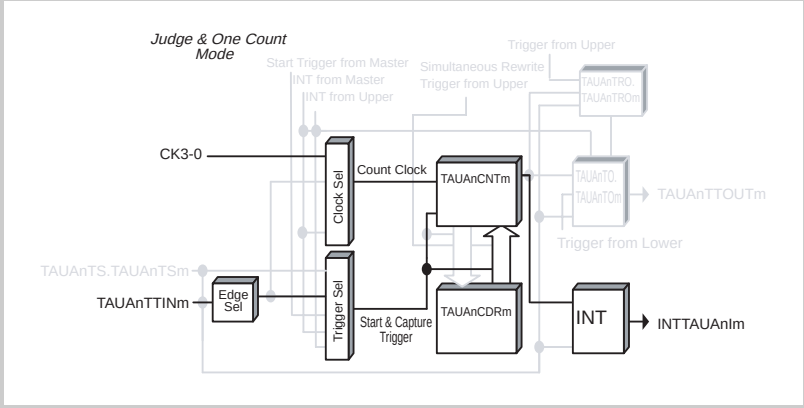


Figure 15-63 Block diagram for TAUAnTTINm input signal width judgment function

The following settings apply to the general timing diagram:

- INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm (TAUAnCMORm.TAUAnMD0 = 0)
- TAUAnTTINm valid start edge = rising edge, TAUAnTTINm valid stop edge = falling edge (TAUAnCMURm.TAUAnTIS[1:0] = 11_B)

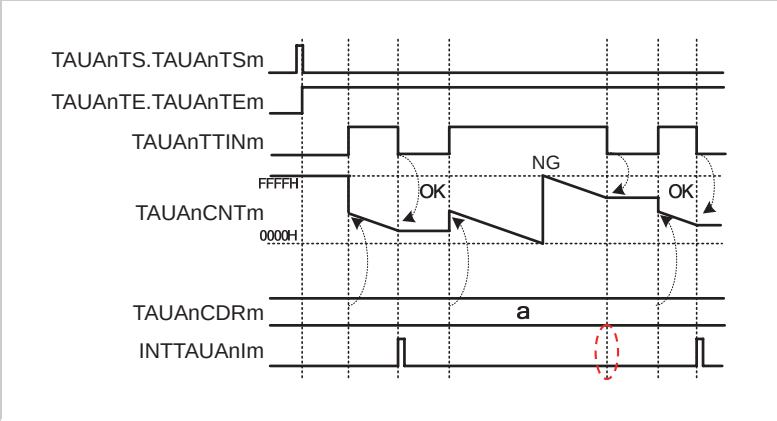


Figure 15-64 General timing diagram for TAUAnTTINm input signal width judgment function

(3) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-59 TAUAnCMORm settings for TAUAnTTINm input signal width judgment function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0111: Judge & one-count mode
TAUAnMD0	0: INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm 1: INTTAUAnIm is generated when TAUAnCNTm > TAUAnCDRm

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-60 TAUAnCMURm settings for TAUAnTTINm input signal width judgment function

Bit name	Setting
TAUAnTIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm input signal width judgment function. Therefore, these registers must be set to 0.

Table 15-61 Simultaneous rewrite settings for TAUAnTTINm input signal width judgment function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(4) Operating procedure for TAUAnTTINm input signal width judgment function

Table 15-62 Operating procedure for TAUAnTTINm input signal width judgment function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-59 "TAUAnCMORm settings for TAUAnTTINm input signal width judgment function" on page 709* and *Table 15-60 "TAUAnCMURm settings for TAUAnTTINm input signal width judgment function" on page 709*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TAUAnTEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a TAUAnTTINm start edge is detected, TAUAnCNTm loads the TAUAnCDRm value.
During operation	Detection of TAUAnTTINm edges The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at any time.	TAUAnCNTm counts down. When a TAUAnTTINm stop edge is detected: • TAUAnCNTm stops and waits for detection of the TAUAnTTINm start edge. • TAUAnCNTm compares the values and judges the condition according to the TAUAnCMORm.TAUAnMD0. • If the condition is satisfied, INTTAUAnIm is generated Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart

15.18 Independent Channel Real-Time Functions

This chapter describes functions that output the value of the TAUAnTRO.TAUAnTROm bit in real time:

- *15.18.1 “Real-time output function type 1”*
- *15.18.2 “Real-time output function type 2”*

15.18.1 Real-time output function type 1

(1) Overview

Summary This function outputs the value of the TAUAnTRO.TAUAnTROm bit from TAUAnTTOUTm when a specified channel generates an interrupt (INTTAUAnIm). In this function, the interrupt is generated at certain specified intervals.

The upper channel generates the real-time output trigger (TAUAnTRC.TAUAnTRCm = 1), and the lower channel receives this trigger and performs real-time output (TAUAnTRC.TAUAnTRCm = 0).

Prerequisites

- One channel using the TAUAnTTOUTm control of one of more other channels
- The operation mode of the upper channel must be set to interval timer mode. See *Table 15-63 "TAUAnCMORm settings for real-time output function type 1" on page 715*.
- The operation mode of the lower channel(s) can be set as desired.
- The channel output mode of all the channels must be set to independent channel output mode 1 with real-time output. See *15.9 "Channel Output Modes" on page 627*.
- Real-time output must be enabled for the upper channel (TAUAnTRE.TAUAnTREm = 1).

Description The counter of the upper channel is started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation. The current value of the data register of the upper channel (TAUAnCDRm) is loaded to the counter (TAUAnCNTm) and the counter starts to count down from this value.

When the counter of the upper channel reaches 0000_H, INTTAUAnIm is generated and TAUAnTTOUTm outputs the current value of the real-time output bit (TAUAnTRO.TAUAnTROm) in all the channels for which TAUAnTRE.TAUAnTREm is set. TAUAnCNTm reloads the TAUAnCDRm value, and then subsequently continues operation.

The TAUAnTTOUTm signal only changes when an interrupt is generated, and then only when its value is different to current value of TAUAnTRO.TAUAnTROm at the moment that the interrupt is generated.

Conditions

- The channel that detects the generation of INTTAUAnIm is specified by setting TAUAnTRC.TAUAnTRCm to 1 for the corresponding channel. The TAUAnTRC.TAUAnTRCm bit must be cleared to 0 for all other channels that do not generate the real-time output trigger.
- If real-time output of a lower channel is disabled (TAUAnTRE.TAUAnTREm = 0) or the channel itself is used as the rewrite trigger (TAUAnTRC.TAUAnTRCm = 1), the value of that channel's TAUAnTRO.TAUAnTROm bit is output when INTTAUAnIm is generated by that channel.
- If real-time output of a lower channel is enabled (TAUAnTRE.TAUAnTREm = 1) and TAUAnTRC.TAUAnTRCm = 0, the value of that channel's TAUAnTRO.TAUAnTROm bit is output when INTTAUAnIm is generated by an upper channel.
- If the TAUAnCMORm.TAUAnMD0 bit is set to 0, the first interrupt after a start or restart is not output. For details, see *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation When Counter Starts or Restarts" on page 639*.

(2) Equations

INTTAUAnIm generation cycle = count clock cycle × (TAUAnCDRm value + 1)

(3) Block diagram and general timing diagram

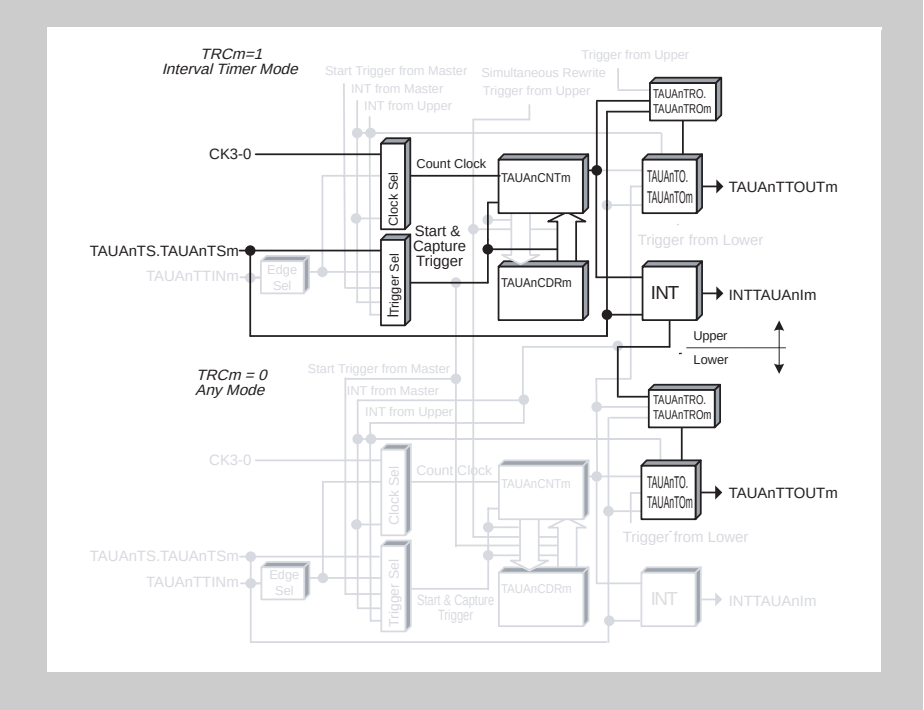


Figure 15-65 Block diagram for real-time output function type 1

The following settings apply to the general timing diagram:

- INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)

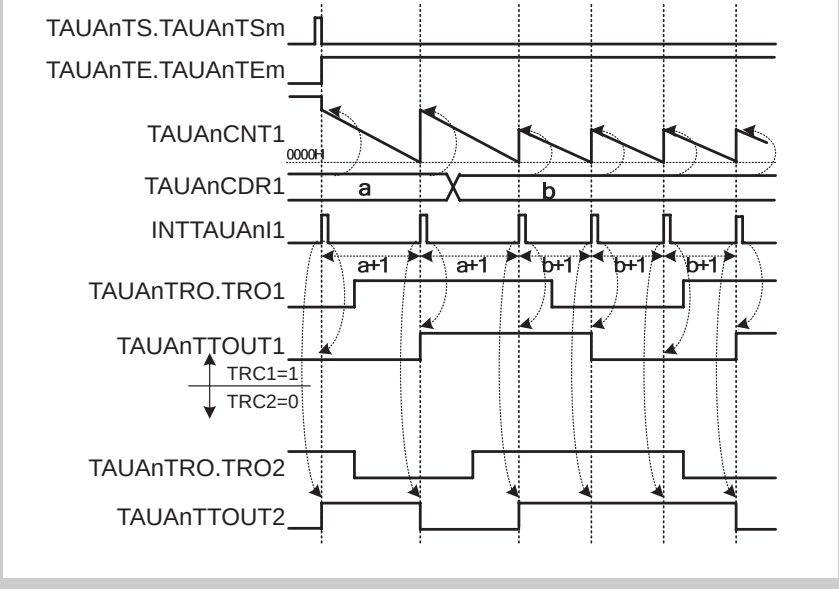


Figure 15-66 General timing diagram for real-time output function type 1

(4) Register settings for the upper channel

(a) TAUAnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-63 TAUAnCMORm settings for real-time output function type 1

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm is not generated at operation start. 1: INTTAUAnIm is generated at operation start.

(b) TAUAnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-64 TAUAnCMURm settings for real-time output function type 1

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode for the upper channel

Table 15-65 Control bit settings for independent channel output mode 1 with real-time output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	1: Enables real-time output
TAUAnTRO.TAUAnTROm	0: Real-time output is low 1: Real-time output is high
TAUAnTRC.TAUAnTRCm	1: Channel m generates its own real-time trigger
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the upper channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the real-time output function type 1. Therefore, these registers must be set to 0.

Table 15-66 Simultaneous rewrite settings for real-time output function type 1

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(5) Register settings for the lower channel(s)

(a) TAUAnCMORm for the lower channel(s)

The TAUAnCMORm register of the lower channel(s) can be set arbitrarily.

(b) TAUAnCMURm for the lower channel(s)

The TAUAnCMURm register of the lower channel(s) can be set arbitrarily.

(c) Channel output mode for the lower channel(s)

Table 15-67 Control bit settings for the lower channel in independent channel output mode 1 with real-time output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TTAUAnOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTRE.TAUAnTREm	1: Enables real-time output
TAUAnTRO.TAUAnTROm	0: Real-time output is low 1: Real-time output is high
TAUAnTRC.TAUAnTRCm	0: The next upper channel generates the real-time trigger for channel m
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the lower channel(s)

The simultaneous rewrite registers of the lower channel(s) can be set arbitrarily.

(6) Operating procedure for real-time output function type 1

Table 15-68 Operating procedure for real-time output function type 1

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *Table 15-63 "TAUAnCMORm settings for real-time output function type 1" on page 715* and *Table 15-64 "TAUAnCMURm settings for real-time output function type 1" on page 715*. Set the TAUAnCMORm register and TAUAnCMURm registers for the lower channel as described in *(5) "Register settings for the lower channel(s)" on page 717*. Set the value of the TAUAnCDRm register. (Only for the channels for which TAUAnTRC.TAUAnTRCm is set.) Set the channel output mode for the upper channel by setting the control bits as described in *Table 15-65 "Control bit settings for independent channel output mode 1 with real-time output" on page 716*. Set the channel output mode for the lower channel by setting the control bits as described in *Table 15-67 "Control bit settings for the lower channel in independent channel output mode 1 with real-time output" on page 717*.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm = 1 in the channel where TAUAnTRC.TAUAnTRCm is set to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	[Channel where TAUAnTRC.TAUAnTRCm is set to 1] TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. INTTAUAnIm is generated when TAUAnCMORm.TAUAnMD0 is set to 1.
During operation	The TAUAnCDRm register and TAUAnTRO.TAUAnTROm can be changed at any time. The TAUAnCNTm register can be read at any time.	TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value, and then continues count operation. • INTTAUAnIm is generated. • TAUAnTTOUTm outputs the current value of the real-time output bit TAUAnTRO.TAUAnTROm. The operation is then repeated
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnTTOUTm retain their current values.

Restart

(7) Specific timing diagrams

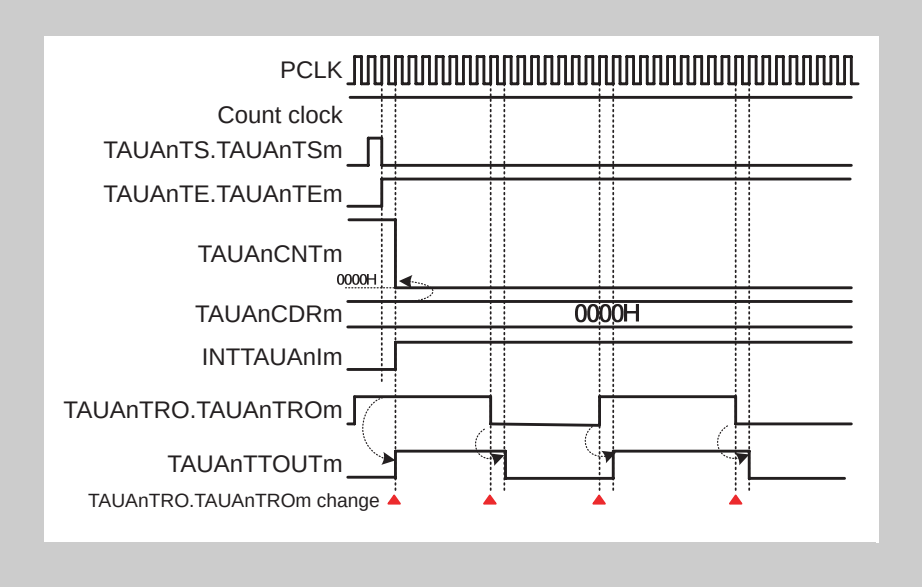


Figure 15-67 TAUAnCDRm = 0000_H, TAUAnCMORm.TAUAnMD0 = 1

- The value of TAUAnTTOUTm follows the TAUAnTRO.TAUAnTROm settings with once PCLK cycle delay.

15.18.2 Real-time output function type 2

(1) Overview

Summary

This function outputs the value of the TAUAnTRO.TAUAnTROm bit from TAUAnTTOUTm when a specified channel generates an interrupt (INTTAUAnIm). In this function, the interrupt is generated when a valid TAUAnTTINm input edge is detected or the function starts.

The upper channel generates the real-time output trigger (TAUAnTRC.TAUAnTRCm = 1), and the lower channel receives this trigger and performs real-time output (TAUAnTRC.TAUAnTRCm = 0).

Prerequisites

- One channel using the TAUAnTTOUTm control of one of more other channels
- The operation mode of the upper channel must be set to capture mode. See *Table 15-69 "TAUAnCMORm settings for real-time output function type 2" on page 722*.
- The operation mode of the lower channel(s) can be set as desired.
- The channel output mode of all the channels must be set to independent channel output mode 1 with real-time output. See *15.9 "Channel Output Modes" on page 627*.
- Real-time output must be enabled for the upper channel (TAUAnTRE.TAUAnTREm = 1).

Description

The counter of the upper channel is started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation. The counter of the upper channel starts to count up.

When a valid TAUAnTTINm input edge is generated on the upper channel, an interrupt is generated, and TAUAnTTOUTm outputs the current value of the real-time output bit (TAUAnTRO.TAUAnTROm) on all channels (only for channels for which TAUAnTRE.TAUAnTREm = 1).

The TAUAnTTOUTm signal only changes when an interrupt is generated, and then only when its value is different to current value of TAUAnTRO.TAUAnTROm at the moment that the interrupt is generated.

Conditions

- The channel that detects the generation of INTTAUAnIm is specified by setting TAUAnTRC.TAUAnTRCm to 1 for the corresponding channel. The TAUAnTRC.TAUAnTRCm bit must be cleared to 0 for all other channels that do not generate the real-time output trigger.
- If real-time output of a lower channel is disabled (TAUAnTRE.TAUAnTREm = 0) or the channel itself is used as the rewrite trigger (TAUAnTRC.TAUAnTRCm = 1), the value of that channel's TAUAnTRO.TAUAnTROm bit is output when INTTAUAnIm is generated by that channel.
- If real-time output of a lower channel is enabled (TAUAnTRE.TAUAnTREm = 1) and TAUAnTRC.TAUAnTRCm = 0, the value of that channel's TAUAnTRO.TAUAnTROm bit is output when INTTAUAnIm is generated by an upper channel.
- If the TAUAnCMORm.TAUAnMD0 bit is set to 0, the first interrupt after a start or restart is not output. For details, see *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation When Counter Starts or Restarts" on page 639*.

(2) Block diagram and general timing diagram

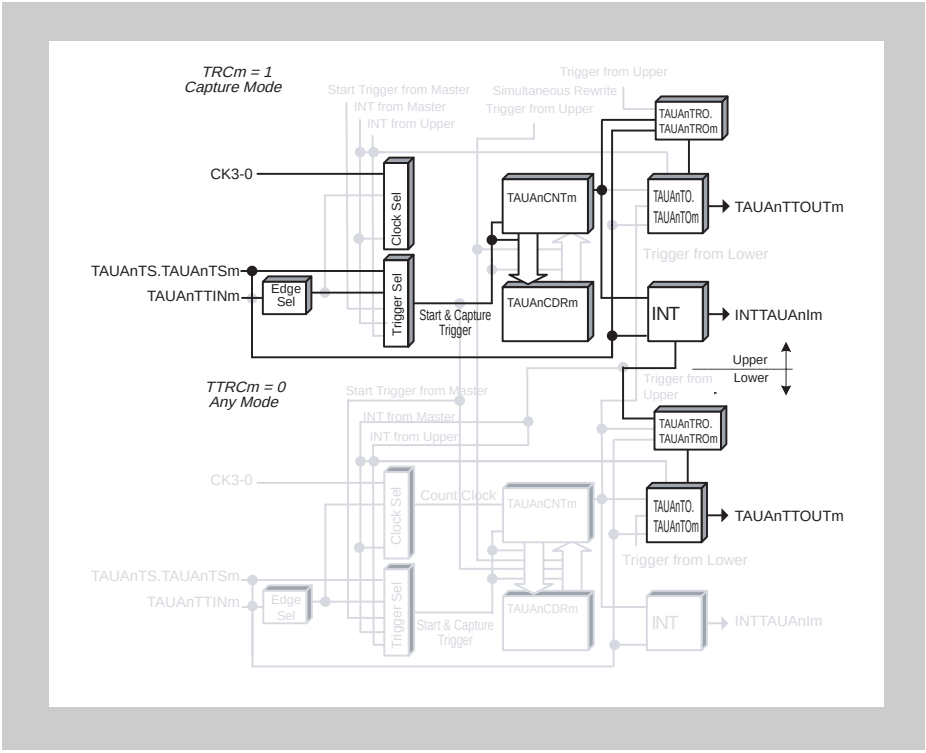


Figure 15-68 Block diagram for real-time output function type 2

The following settings apply to the general timing diagram:

- INTTAUAnIm not output at operation start (TAUAnCMORm.TAUAnMD0 = 0)

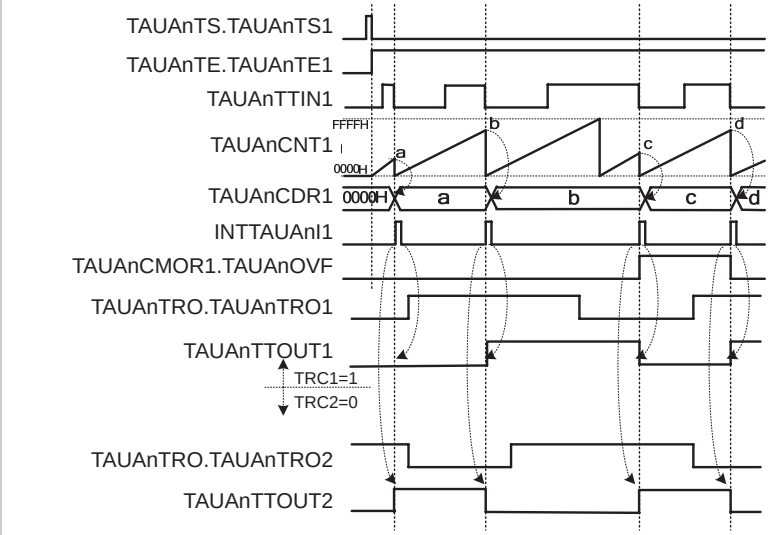


Figure 15-69 General timing diagram for real-time output function type 2

(3) Register settings for the upper channel

(a) TAUAnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-69 TAUAnCMORm settings for real-time output function type 2

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
<R> TAUAnSTS[2:0]	001: Valid edge of the TAUAnTTINm input signal is the external start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0010: Capture mode
TAUAnMD0	0: INTTAUAnIm is not generated at operation start. 1: INTTAUAnIm is generated at operation start.

(b) TAUAnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-70 TAUAnCMURm settings for real-time output function type 2

Bit name	Setting
<R> <R> TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the upper channel

Table 15-71 Control bit settings for independent channel output mode 1 with real-time output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	1: Enables real-time output
TAUAnTRO.TAUAnTROm	0: Real-time output is low 1: Real-time output is high
TAUAnTRC.TAUAnTRCm	1: Channel m generates its own real-time trigger
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the upper channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the real-time output function type 2. Therefore, these registers must be set to 0.

Table 15-72 Simultaneous rewrite settings for real-time output function type 2

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(4) Register settings for the lower channel(s)

(a) TAUAnCMORm for the lower channel(s)

The TAUAnCMORm register of the lower channel(s) can be set arbitrarily.

(b) TAUAnCMURm for the lower channel(s)

The TAUAnCMURm register of the lower channel(s) can be set arbitrarily.

(c) Channel output mode for the lower channel(s)

Table 15-73 Control bit settings for lower channel in independent channel output mode 1 with real-time output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output 1: Enables real-time output
TAUAnTRO.TAUAnTROm	0: Real-time output is low 1: Real-time output is high
TAUAnTRC.TAUAnTRCm	0: The upper channel generates the real-time trigger for channel m 1: Channel m generates its own real-time trigger
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the lower channel(s)

The simultaneous rewrite registers of the lower channel(s) can be set arbitrarily.

(5) Operating procedure for real-time output function type 2

Table 15-74 Operating procedure for real-time output function type 2

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *Table 15-69 "TAUAnCMORm settings for real-time output function type 2" on page 722* and *Table 15-70 "TAUAnCMURm settings for real-time output function type 2" on page 722*. Set the TAUAnCMORm register and TAUAnCMURm registers for the lower channel as described in *(4) "Register settings for the lower channel(s)" on page 724*. Set the channel output mode for the upper channel by setting the control bits as described in *Table 15-71 "Control bit settings for independent channel output mode 1 with real-time output" on page 723*. Set the channel output mode for the lower channel by setting the control bits as described in *Table 15-73 "Control bit settings for lower channel in independent channel output mode 1 with real-time output" on page 724*.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm of the channel where TAUAnTRC.TAUAnTRCm is set to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	[Channel where TAUAnTRC.TAUAnTRCm is set to 1] TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm is cleared to 0000_H. When TAUAnCMORm.TAUAnMD0 is set to 1, INTTAUAnIm is generated.
During operation	TAUAnTRO.TAUAnTROm can be changed at any time.	TAUAnCNTm starts to count up from 0000_H. When a TAUAnTTINm input valid edge is detected: • INTTAUAnIm is generated. • TAUAnCSRm.TAUAnOVF is cleared to 0. TAUAnTTOUTm outputs the current value of the real-time output bit TAUAnTRO.TAUAnTROm. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it, TAUAnCSRm.TAUAnOVF, and TAUAnTTOUTm retain their current values.

Restart

15.19 Independent Channel Simultaneous Rewrite Functions

This chapter describes functions that carry out simultaneous rewrite:

- *15.19.1 “Simultaneous rewrite trigger generation function type 1”*
- *15.19.2 “Simultaneous rewrite trigger generation function type 2”*

15.19.1 Simultaneous rewrite trigger generation function type 1

(1) Overview

Summary This function generates an interrupt on a specific channel that can be used by lower channels as a simultaneous rewrite trigger. The interrupt is generated at regular intervals.

The upper channel generates the simultaneous rewrite trigger (TAUAnRDC.TAUAnRDCm = 1), and the lower channel receives this trigger and performs simultaneous rewriting (TAUAnRDC.TAUAnRDCm = 0).

Prerequisites

- At least two channels lower than a channel used as an upper channel, each with simultaneous rewriting enabled (TAUAnRDE.TAUAnRDEm = 1)
- The operation mode of the upper channel must be set to interval timer mode. See *Table 15-75 "TAUAnCMORm settings for simultaneous rewrite trigger generation function type 1" on page 730*.
- For details about which operation modes can be specified for lower channels, see *Table 15-10 "Simultaneous rewrite methods and when they are triggered" on page 615*.
- TAUAnTTOUTm is not used for any channel in this function.

Description The counter is enabled by setting the channel trigger bit of the upper channel or lower channel (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TEm to 1, enabling counting. The current value of the data register buffer of the upper channel (TAUAnCDRm buf) is written to the counter (TAUAnCNTm) and the counter starts to count down from this value. The counter(s) of the lower channel(s) start to count as specified by their selected operating modes.

When a counter reaches 0000_H, an interrupt is generated from the channel. The TAUAnCNTm loads the current value of the corresponding TAUAnCDRm buffer, and then subsequently continues operation.

If the channel where the interrupt occurs is specified as the trigger channel for simultaneous rewrite (TAUAnRDC.TAUAnRDCm = 1) and is an upper channel, simultaneous rewrite takes place on all lower channels in which simultaneous rewrite is currently possible (TAUAnRSF.TAUAnRSFm = 1).

The values of the data registers are copied to the corresponding data register buffers. Each time a counter starts to count down, it reads the value in the data register buffer and counts down from this value.

The value of a data register can be changed at any time, but it is only transferred to the corresponding data register buffer when simultaneous rewrite occurs.

Conditions

- The channel that detects the generation of INTTAUAnIm is specified by setting TAUAnRDC.TAUAnRDCm to 1 for the corresponding channel. The TAUAnRDC.TAUAnRDCm bit must be 0 for all other channels in which simultaneous rewrite should take place.
- If the TAUAnCMORm.TAUAnMD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details, see *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation When Counter Starts or Restarts" on page 639*.

(2) Equations

Simultaneous rewrite trigger generation cycle = count clock cycle × (TAUAnCDRm + 1)

To control simultaneous rewrite, the following condition must be satisfied:

For PMW:

TAUAnCDRm = [(value of TAUAnCDRm of master channel subject to simultaneous rewrite + 1) × number of interrupts] - 1

For triangle PWM:

TAUAnCDRm = [(value of TAUAnCDRm of master channel subject to simultaneous rewriting + 1) × 2 × number of interrupts] - 1

That is, the ratio of TAUAnCDRm + 1 and TAUAnCDRm_master + 1 must be an integer. This integer corresponds to the number of interrupts.

Note that, for triangle PWM, the cycle is doubled.

(3) Block diagram and general timing diagram

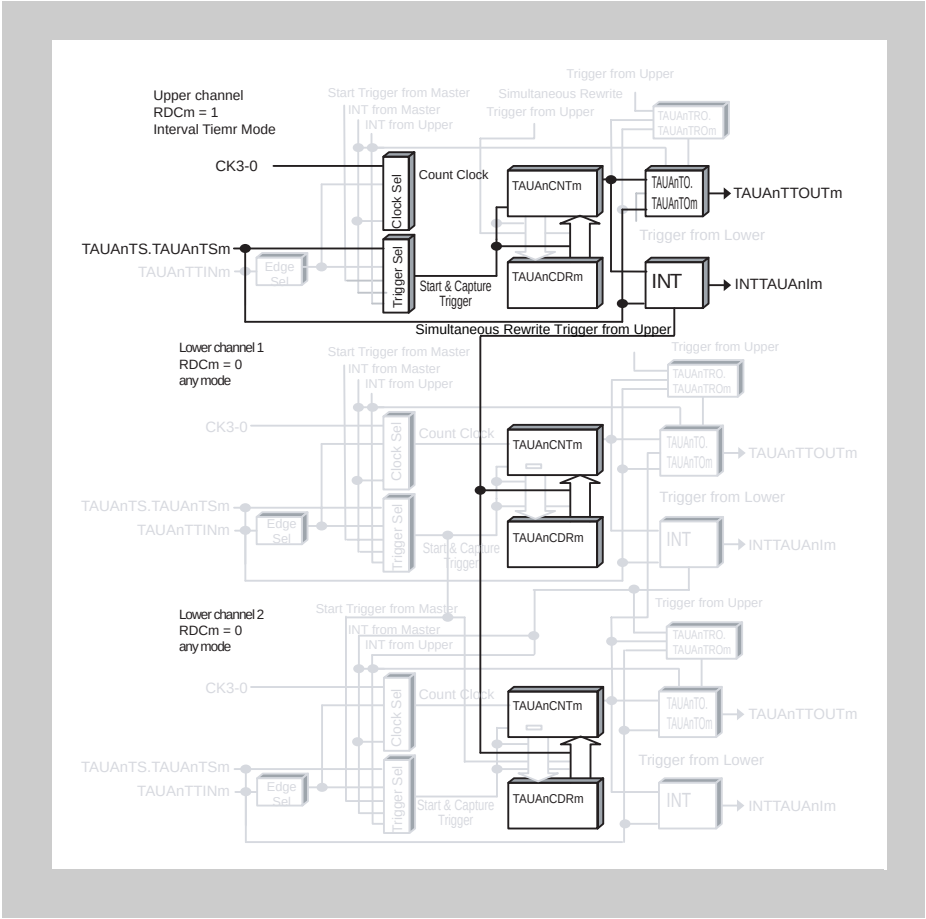


Figure 15-70 Block diagram for simultaneous rewrite trigger generation function type 1

The following settings apply to the general timing diagram:

- INTTAUAnIm generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)

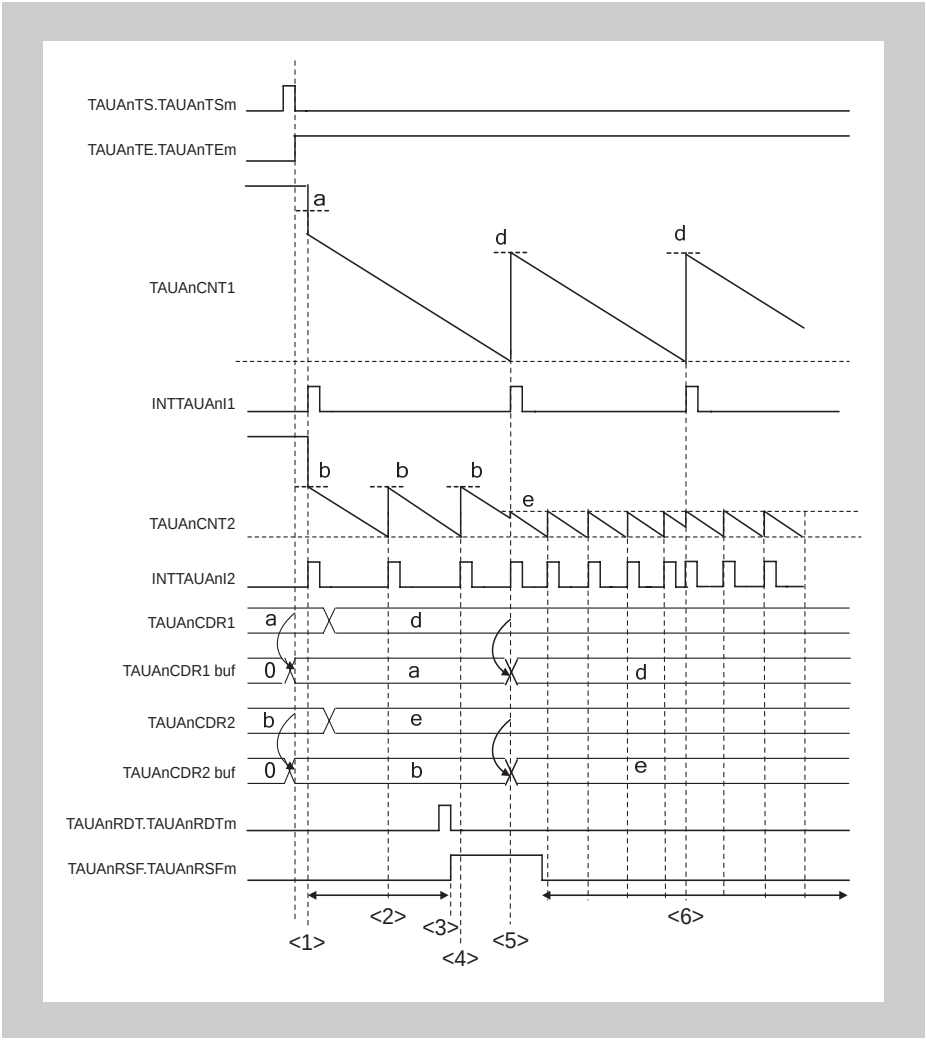


Figure 15-71 General timing diagram for simultaneous rewrite trigger generation function type 1

Description:

1. When TAUAnTS.TAUAnTSm is set to 1, the value of TAUAnCDRm is copied to the TAUAnCDRm buffer.
2. The TAUAnCDRm register can always be written to.
3. The reload data trigger bit (TAUAnRDT.TAUAnRDTm) is set to 1, which sets the status flag (TAUAnRSF.TAUAnRSFm = 1), enabling simultaneous rewriting.
4. Even though simultaneous rewriting is enabled, it does not take place because it is only triggered by an interrupt on channel 1.
5. Simultaneous rewriting is triggered by INT1, which is caused by counter 1 reaching 0000H. The TAUAnCDRm values are loaded to the corresponding TAUAnCDRm buffers.
6. The counter counts down and awaits the next simultaneous rewrite trigger. The values of the TAUAnCDRm registers can be changed again.

(4) Register settings for the upper channel

(a) TAUAnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-75 TAUAnCMORm settings for simultaneous rewrite trigger generation function type 1

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start 1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-76 TAUAnCMURm settings for simultaneous rewrite trigger generation function type 1

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode for the upper channel

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the upper channel

Table 15-77 Simultaneous rewrite settings for simultaneous rewrite trigger generation function type 1

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
TAUAnRDC.TAUAnRDCm	1: Channel is monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger

(5) Register settings for the lower channel(s)

(a) TAUAnCMORm for the lower channel(s)

For the TAUAnCMORm registers of lower channels, use the TAUAnCMORm register settings of operation modes that can be specified. (For details, see *Table 15-10 “Simultaneous rewrite methods and when they are triggered” on page 615*.)

(b) TAUAnCMURm for the lower channel(s)

For the TAUAnCMURm registers of lower channels, use the TAUAnCMURm register settings of operation modes that can be specified. (For details, see *Table 15-10 “Simultaneous rewrite methods and when they are triggered” on page 615*.)

(c) Channel output mode for the lower channel(s)

Output is possible according to the operation mode setting (master or slave) of a lower channel.

(d) Simultaneous rewrite for the lower channel(s)

Table 15-78 Simultaneous rewrite settings for the lower channel in simultaneous rewrite trigger generation function type 1

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous-rewrite trigger

(6) Operating procedure for simultaneous rewrite trigger generation function type 1

Table 15-79 Operating procedure for simultaneous rewrite trigger generation function type 1

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *Table 15-75 "TAUAnCMORm settings for simultaneous rewrite trigger generation function type 1" on page 730* and *Table 15-76 "TAUAnCMURm settings for simultaneous rewrite trigger generation function type 1" on page 730*. Set the TAUAnCMORm register and TAUAnCMURm registers for the lower channel as described in *(5) "Register settings for the lower channel(s)" on page 731*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. When TAUAnCMORm.TAUAnMD0 = 1, INTTAUAnIm is generated.
During operation	TAUAnRDT.TAUAnRDTm can be changed. TAUAnRSF.TAUAnRSFm can be read at all times.	TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value, and then continues count operation. • INTTAUAnIm is generated. Simultaneous rewrite is controlled when INTTAUAnIm is generated from the channel where TAUAnRDC.TAUAnRDCm is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains the current value.

Restart →

15.19.2 Simultaneous rewrite trigger generation function type 2

(1) Overview

Summary

This function generates an interrupt on a specific channel that can be used by lower channels as a simultaneous rewrite trigger. The interrupt is triggered by a valid TAUAnTTINm input edge or the function starting.

The upper channel generates the simultaneous rewrite trigger (TAUAnRDC.TAUAnRDCm = 1), and the lower channel receives this trigger and performs simultaneous rewriting (TAUAnRDC.TAUAnRDCm = 0).

Prerequisites

- At least two channels used as lower channels than upper channels, each with simultaneous rewriting enabled (TAUAnRDE.TAUAnRDEm = 1)
- The operation mode of the upper channel must be set to capture mode. See *Table 15-80 “TAUAnCMORm settings for simultaneous rewrite trigger generation function type 2” on page 737*.
- For details about which operation modes can be specified for lower channels, see *Table 15-10 “Simultaneous rewrite methods and when they are triggered” on page 615*.
- The channel output mode of the upper channel must be set to independent channel output mode controlled by software. See *15.9 “Channel Output Modes” on page 627*.
- The channel output mode of the lower channel(s) can be set as desired.

Description

The counter is enabled by setting the channel trigger bit of the upper channel or lower channel (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TEm to 1, enabling counting. The counter of the upper channel starts to count up, and the counter(s) of the lower channel(s) start to count as specified by their selected operating modes.

When a valid TAUAnTTINm input edge is generated on the upper channel, an interrupt is generated, and an interrupt is generated upon trigger detection on the lower channel.

TAUAnRDC.TAUAnRDCm = 1 on the upper channel, therefore simultaneous rewrite takes place on all lower channels in which simultaneous rewrite is currently possible (TAUAnRSF.TAUAnRSFm = 1).

The values of the data registers are copied to the corresponding data register buffers.

The value of a data register can be changed at any time, but it is only transferred to the corresponding data register buffer when simultaneous rewrite occurs.

Conditions

- The channel that detects the generation of INTTAUAnIm is specified by setting TAUAnRDC.TAUAnRDCm to 1 for the corresponding channel. The TAUAnRDC.TAUAnRDCm bit must be 0 for all other channels on which simultaneous rewriting must take place.
- If the TAUAnCMORm.TAUAnMD0 bit is set to 1, an interrupt is generated when the function starts. For details, see *15.11 “TAUAnTTOUTm Output and INTTAUAnIm Generation When Counter Starts or Restarts” on page 639*.

(2) Block diagram and general timing diagram

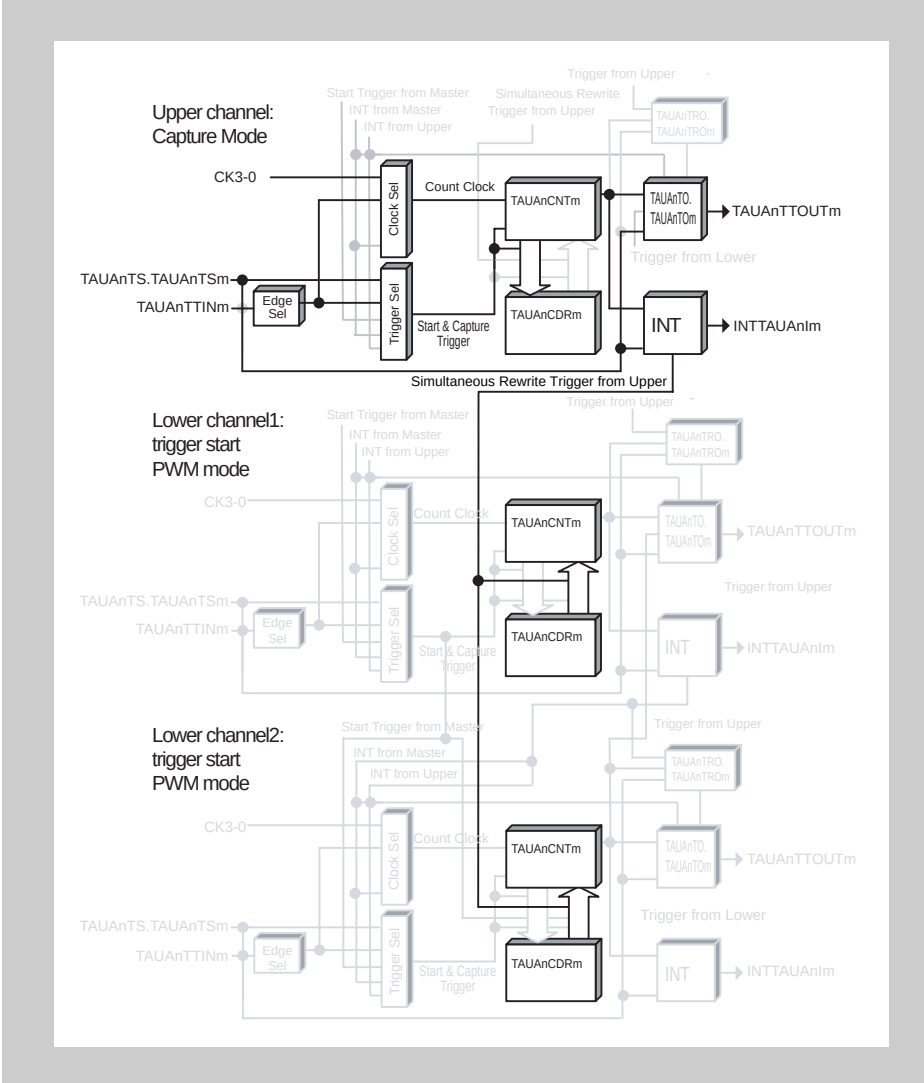


Figure 15-72 Block diagram for simultaneous rewrite trigger generation function type 2

The following settings apply to the general timing diagram:

- INTTAUAnIm not generated at operation start (TAUAnCMORm.TAUAnMD0 = 0)
- Falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)
- Upper channel (channel 1) generates simultaneous rewrite trigger

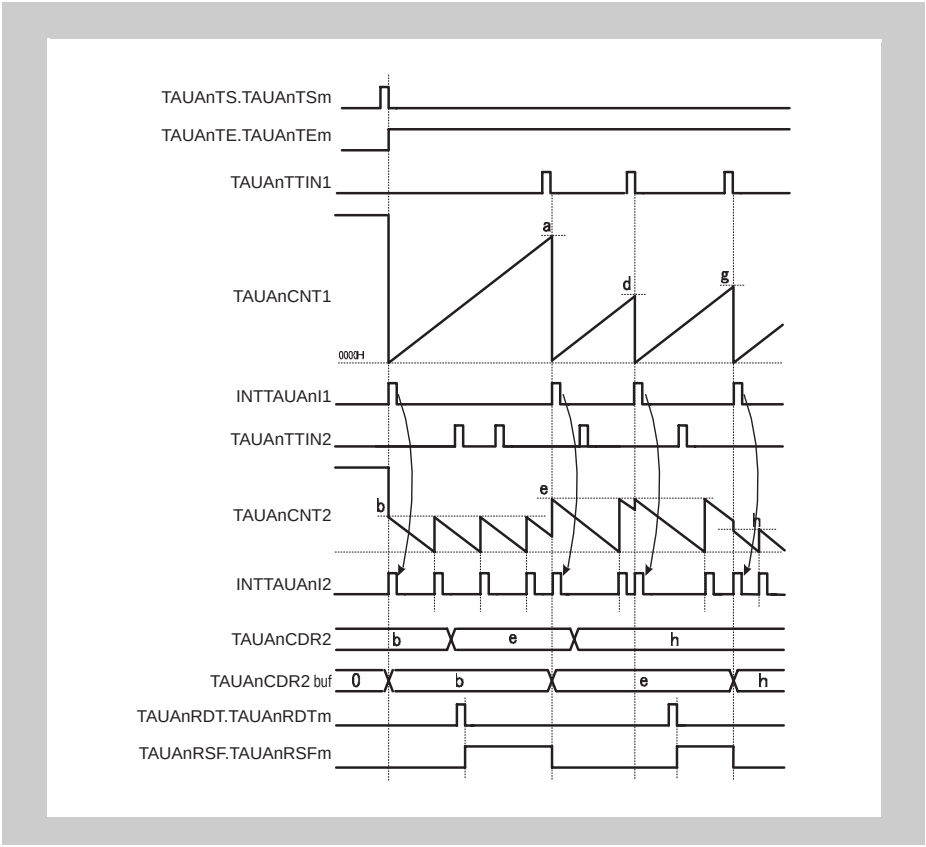


Figure 15-73 General timing diagram for simultaneous rewrite trigger generation function type 2

(3) Register settings for the upper channel

(a) TAUAnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-80 TAUAnCMORm settings for simultaneous rewrite trigger generation function type 2

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	001:Valid edge of the TAUAnTTINm input signal is the external capture trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0010: Capture mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start 1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-81 TAUAnCMURm settings for simultaneous rewrite trigger generation function type 2

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the upper channel

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the upper channel

Table 15-82 Simultaneous rewrite settings for simultaneous rewrite trigger generation function type 2

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
TAUAnRDC.TAUAnRDCm	1: Channel is monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger

(4) Register settings for the lower channel(s)

(a) TAUAnCMORm for the lower channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAn MD[4:1]				TAUAn MD0

Table 15-83 TAUAnCMORm settings for simultaneous rewrite trigger generation function type 2

Bit name	Setting
TAUAnCKS[1:0]	00:Prescaler output CK0 01:Prescaler output CK1 10:Prescaler output CK2 11:Prescaler output CK3 The value of the TAUAnCKS[1:0] bits of the master and slave channels must be the same.
TAUAnCCS[1:0]	00:Use the operation clock as the count clock.
TAUAnMAS	1: The channel is the master channel.
TAUAnSTS[2:0]	001:Use the valid TAUAnTTINm input edge signal as the start trigger.
TAUAnCOS[1:0]	00:These are not used, so set them to 00.
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the lower channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-84 TAUAnCMURm settings for simultaneous rewrite trigger generation function type 2

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the lower channel(s)

Output is possible according to the trigger start PWM mode setting.

(d) Simultaneous rewrite for the lower channel(s)

Table 15-85 Simultaneous rewrite settings for the lower channel in simultaneous rewrite trigger generation function type 2

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous-rewrite trigger

(5) Operating procedure for simultaneous rewrite trigger generation function type 2

Table 15-86 Operating procedure for simultaneous rewrite trigger generation function type 2

		Operation	Status of TAUAn
	Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *Table 15-80 "TAUAnCMORm settings for simultaneous rewrite trigger generation function type 2" on page 737* and *Table 15-81 "TAUAnCMURm settings for simultaneous rewrite trigger generation function type 2" on page 737*. Set up the TAUAnCMORm and TAUAnCMURm registers of the lower channel as shown in *Table 15-83 "TAUAnCMORm settings for simultaneous rewrite trigger generation function type 2" on page 738* and *Table 15-84 "TAUAnCMURm settings for simultaneous rewrite trigger generation function type 2" on page 739*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Restart	Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm is cleared to 0000_H. INTTAUAnIm is generated when TAUAnCMORm.TAUAnMD0 is set to 1.
	During operation	TAUAnRDT.TAUAnRDTm register can be set at any time. TAUAnRSF.TAUAnRSFm register can be read at any time.	TAUAnCNTm counts up from 0000_H. When a TAUAnTTINm valid edge is detected: • TAUAnCNTm transfers (captures) its value to TAUAnCDRm and returns to 0000_H. • INTTAUAnIm is generated. Simultaneous rewrite is controlled when INTTAUAnIm is generated from the channel where TAUAnRDC.TAUAnRDCm is set to 1. Afterwards, this procedure is repeated.
	Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains the current value.

15.20 Independent Channel One-Phase PWM Function

This chapter describes the One-Phase PWM Function:

- *15.20.1 “One-phase PWM output function”*

15.20.1 One-phase PWM output function

(1) Overview

Summary This function adds dead time to a TAUAnTTINm input signal. The resulting PWM signal is output via TAUAnTTOUTm of the channel and TAUAnTTOUTm of (an) upper channel(s).

Prerequisites

- Two (or more) channels, each with dead time control enabled (TAUAnTDE.TAUAnTDEm = 1)
- The operation mode of the lower channel must be set to one-count mode. See *Table 15-87 "TAUAnCMORm settings for one-phase PWM output function" on page 745*.
- The operation mode of the upper channel(s) can be set as desired.
- The channel output mode of the upper and lower channels must be set to synchronous channel output mode 2 with one-phase PWM output. See *15.9 "Channel Output Modes" on page 627*.

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input start edge is detected. The value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from the TAUAnCDRm value.

When the counter reaches 0000_H an interrupt is generated. The counter returns to $FFFF_H$ and awaits the next valid TAUAnTTINm input start edge.

Conditions

- The TAUAnCMURm.TAUAnTIS[1:0] bits specify the type of width measurement:
  - TAUAnCMURm.TAUAnTIS[1:0] = 10B:
    Detect both rising and falling edges as valid (low-width measurement).
  - TAUAnCMURm.TAUAnTIS[1:0] = 11B:
    Detect both rising and falling edges as valid (high-width measurement).
- The TAUAnTDL.TAUAnTDLm bit specifies the behavior of TAUAnTTOUTm for each channel when an interrupt or valid TAUAnTTINm edge is detected on the lower channel:
  - If TAUAnTDL.TAUAnTDLm = 0, an interrupt is the trigger for setting TAUAnTTOUTm, and a valid TAUAnTTINm edge is the trigger for resetting TAUAnTTOUTm.
  - If TAUAnTDL.TAUAnTDLm = 1, a valid TAUAnTTINm edge is the trigger for setting TAUAnTTOUTm, and an interrupt is the trigger for resetting TAUAnTTOUTm.
- Forced restart is not possible for this function.

(2) Block diagram and general timing diagram

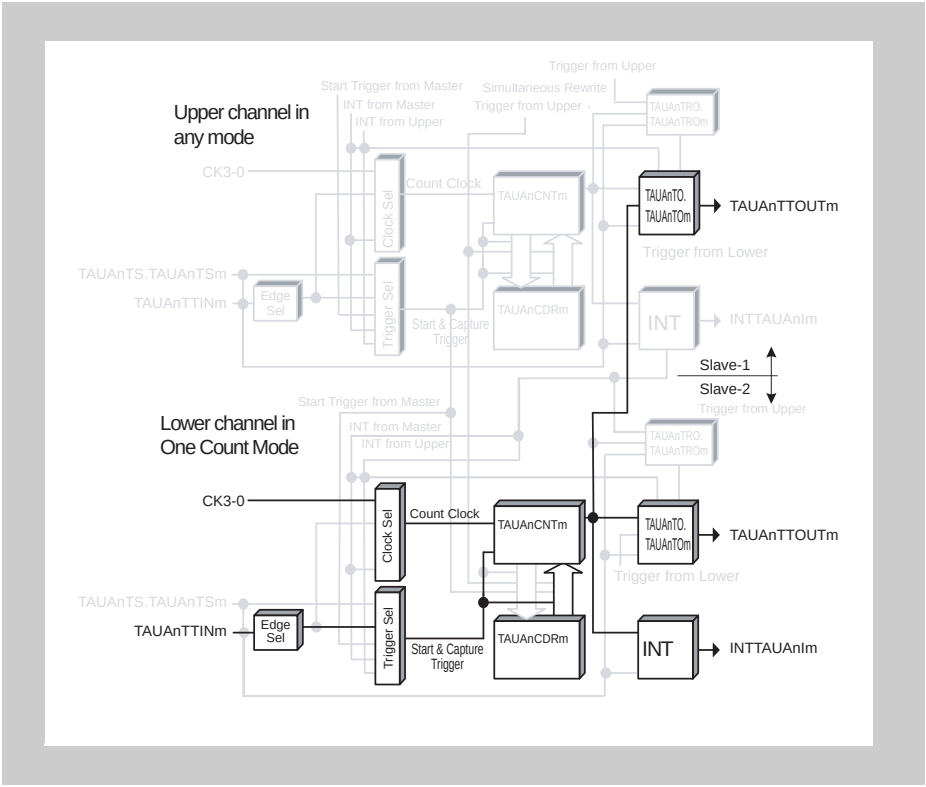


Figure 15-74 Block diagram for one-phase PWM output function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TAUAnTIS[1:0] = 11_B)

For these settings, signals are assumed to be active when at high level.

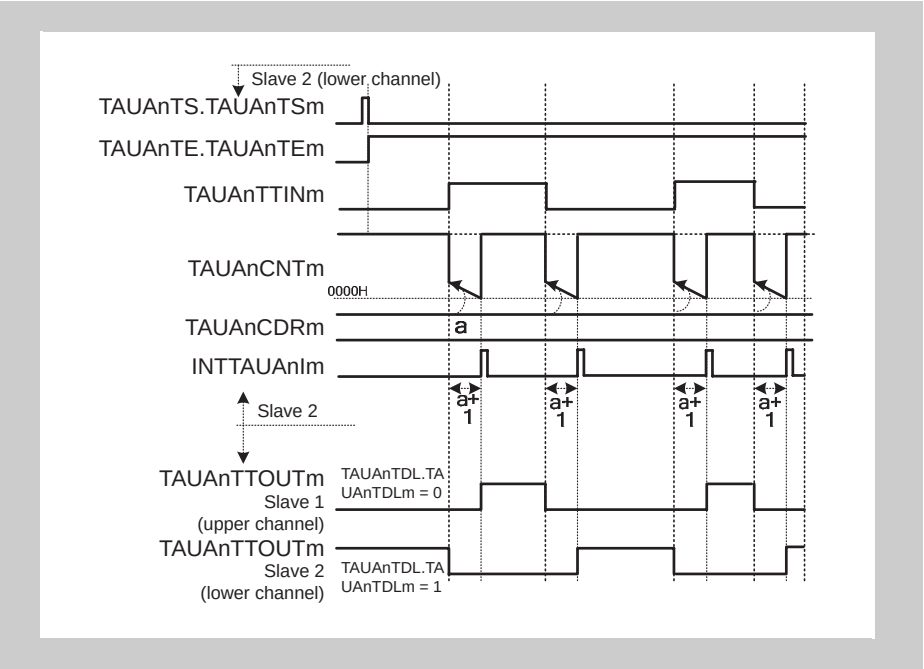


Figure 15-75 General timing diagram for one-phase PWM output function

(3) Register settings for the lower channel

(a) TAUAnCMORm for the lower channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-87 TAUAnCMORm settings for one-phase PWM output function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	001: Valid edge of the TAUAnTTINm input signal is the external start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Enables start trigger detection during counting

(b) TAUAnCMURm for the lower channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-88 TAUAnCMURm settings for one-phase PWM output function

Bit name	Setting
TAUAnTIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode for the lower channel

Table 15-89 Control bit settings for synchronous channel output mode 2 with one-phase PWM output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel output
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	1: Enables dead time operation
TAUAnTDM.TAUAnTDMm	1: Dead time is added upon detection of a TAUAnTTINm input edge from a lower odd channel
TAUAnTDL.TAUAnTDLm	0: An interrupt is the TAUAnTTOUTm set trigger and a valid TAUAnTTINm edge is the TAUAnTTOUTm reset trigger 1: A valid TAUAnTTINm edge is the TAUAnTTOUTm set trigger and an interrupt is the TAUAnTTOUTm reset trigger
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

Caution For TAUAnTDL.TAUAnTDLm, specify the setting that is opposite that of the upper channel.

(d) Simultaneous rewrite for the lower channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the one-phase PWM output function. Therefore, these registers must be set to 0.

Table 15-90 Simultaneous rewrite settings for one-phase PWM output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(4) Register settings for the upper channel(s)

(a) TAUAnCMORm for the upper channel(s)

The TAUAnCMORm register of the upper channel(s) can be set arbitrarily.

(b) TAUAnCMURm for the upper channel(s)

The TAUAnCMURm register of the upper channel(s) can be set arbitrarily.

(c) Channel output mode for the upper channel(s)

Table 15-91 Control bit settings for upper channel(s) for synchronous channel output mode 2 with one-phase PWM output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel output
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	1: Enables dead time operation
TAUAnTDM.TAUAnTDMm	1: Dead time is added upon detection of a TAUAnTTINm input edge from a lower odd channel
TAUAnTDL.TAUAnTDLm	0: An interrupt on the lower channel is the TAUAnTTOUTm set trigger and a valid TAUAnTTINm edge on the lower channel is the TAUAnTTOUTm reset trigger 1: A valid TAUAnTTINm edge on the lower channel is the TAUAnTTOUTm set trigger and an interrupt on the lower channel is the TAUAnTTOUTm reset trigger
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

Caution For TAUAnTDL.TAUAnTDLm, specify the setting that is opposite that of the lower channel.

(d) Simultaneous rewrite for the upper channel(s)

The simultaneous rewrite registers of the upper channel(s) can be set arbitrarily.

(5) Operating procedure for one-phase PWM output function

Table 15-92 Operating procedure for one-phase PWM output function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the lower channel as described in *Table 15-87 "TAUAnCMORm settings for one-phase PWM output function" on page 745* and *Table 15-88 "TAUAnCMURm settings for one-phase PWM output function" on page 745*. Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *(4) "Register settings for the upper channel(s)" on page 747*. Set the value of the TAUAnCDRm register. Set the channel output mode for the upper and lower channel by setting the control bits as described in *Table 15-89 "Control bit settings for synchronous channel output mode 2 with one-phase PWM output" on page 746*.	Channel operation is stopped.
Start operation	Set TAUAnTOE.TAUAnTOEm (slave channels 1 and 2) to 1 (at operation restart only). Set TAUAnTS.TAUAnTSm = 1 for slave channel 2. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TAUAnTEm is set to 1 (slave channel 2) and TAUAnCNTm waits for TAUAnTTINm start edge detection. TAUAnCNTm loads the TAUAnCDRm value.
During operation	The value of the TAUAnCDRm register can be changed at any time. The TAUAnCNTm register can be read at any time.	TAUAnCNTm of slave channel 2 counts down. When it reaches 0000_H: • INTTAUAnIm is generated. • TAUAnCNTm stops counting. The TAUAnTTOUTm level is changed through the TAUAnTTINm edge detection signal and the INTTAUAnIm signal from slave channel 2 to output a one-phase PWM waveform with dead time. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm = 1 for slave channel 2. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains the current value.

Restart

15.21 Other Independent Channel Functions

This chapter describes a function that generates an interrupt when a certain number of TAUAnTTINm pulses has occurred, a function that divides the frequency of TAUAnTTINm, and a function that measures the duration between the function start and a TAUAnTTINm input signal:

- *15.21.1 “External event count function”*
- *15.21.2 “Clock divide function”*
- *15.21.3 “TAUAnTTINm input position detection function”*

15.21.1 External event count function

(1) Overview

Summary This function is used as an event timer. It generates an interrupt (INTTAUAnIm) when a specific number of TAUAnTTINm input pulses has occurred.

Prerequisites
- The operation mode must be set to event count mode. See *Table 15-93 "TAUAnCMORm settings for external event count function" on page 752*.
- TAUAnTTOUTm is not used for this function.

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. When the counter starts, the current value of TAUAnCDRm is loaded to TAUAnCNTm.

When a valid TAUAnTTINm input edge is detected, the value of TAUAnCNTm reduces by 1. TAUAnCNTm retains this value until a valid TAUAnTTINm input edge is detected or the counter is restarted.

When the counter value reaches 0000_H, INTTAUAnIm is generated. TAUAnCNTm loads the TAUAnCDRm value, and then subsequently continues operation.

The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1, which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm stops and retains its value. The counter can be restarted by setting TAUAnTS.TAUAnTSm to 1. The counter can also be restarted without stopping it first (forced restart) by setting TAUAnTS.TAUAnTSm to 1 during operation.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUAnCMURm.TAUAnTIS[1:0] bits:
- If TAUAnCMURm.TAUAnTIS[1:0] = 00_B, the falling edges are counted.
- If TAUAnCMURm.TAUAnTIS[1:0] = 01_B, the rising edges are counted.
- If TAUAnCMURm.TAUAnTIS[1:0] = 10_B, the rising and falling edges are counted.

(2) Equations

Number of valid edges,
detected before INTTAUAnIm is generated = TAUAnCDRm + 1

(3) Block diagram and general timing diagram

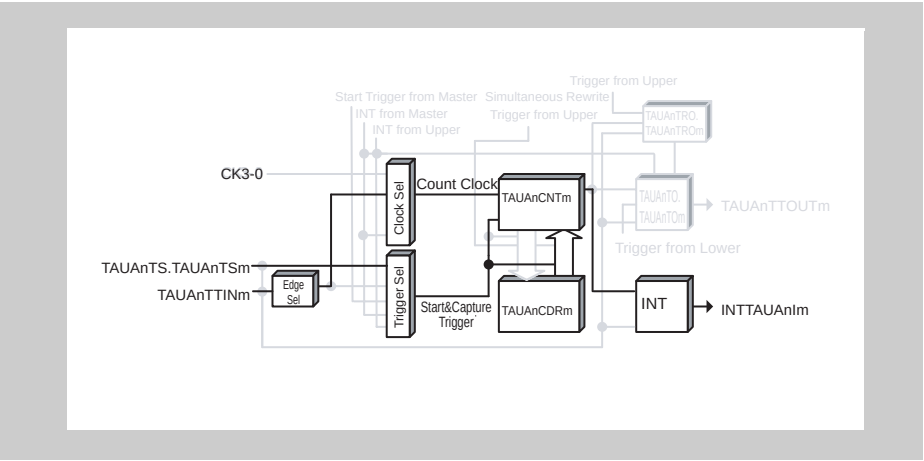


Figure 15-76 Block diagram for external event count function

The following settings apply to the general timing diagram:

- Rising edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 01_B)

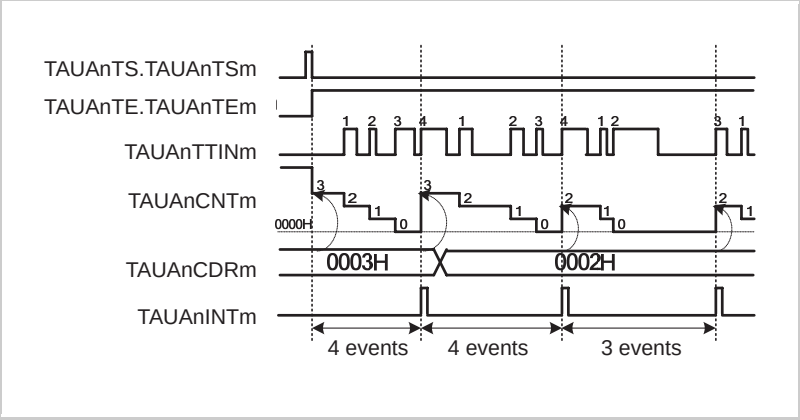


Figure 15-77 General timing diagram for external event count function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-93 TAUAnCMORm settings for external event count function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	01: Valid TAUAnTTINm input edge is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0011: Event count mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-94 TAUAnCMURm settings for external event count function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge 01: Rising edge 10: Rising and falling edge

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the external event count function. Therefore, these registers must be set to 0.

Table 15-95 Simultaneous rewrite settings for external event count function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(5) Operating procedure for external event count function

Table 15-96 Operating procedure for external event count function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-93 "TAUAnCMORm settings for external event count function" on page 752* and *Table 15-94 "TAUAnCMURm settings for external event count function" on page 752*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value, and then waits for detection of the TAUAnTTINm input edge.
During operation	Detection of TAUAnTTINm edges. The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at any tme.	TAUAnCNTm performs count-down operation each time a TAUAnTTINm input edge is detected. When the counter reaches 0000_H: • TAUAnCNTm loads the TAUAnCDRm value, and then continues count operation. • INTTAUAnIm is generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart

(6) Specific timing diagrams

(a) TAUAnCDRm = 0000_H

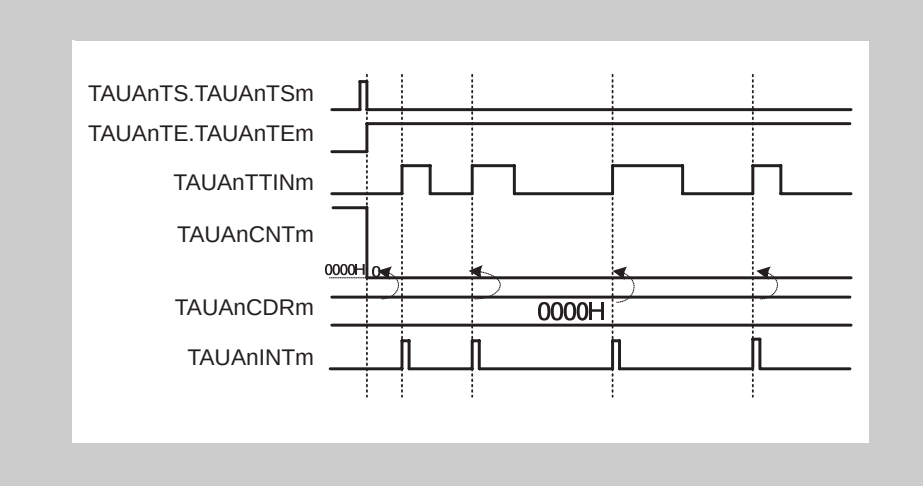


Figure 15-78 TAUAnCDRm = 0000H, TAUAnCMURm.TAUAnTIS[1:0] = 01_B

- If 0000_H = TAUAnCDRm, 0000_H is loaded to TAUAnCNTm every time a valid TAUAnTTINm input edge is detected.

  This means, INTTAUAnIm is generated every time a valid TAUAnTTINm input edge is detected.

(b) Operation stop and restart

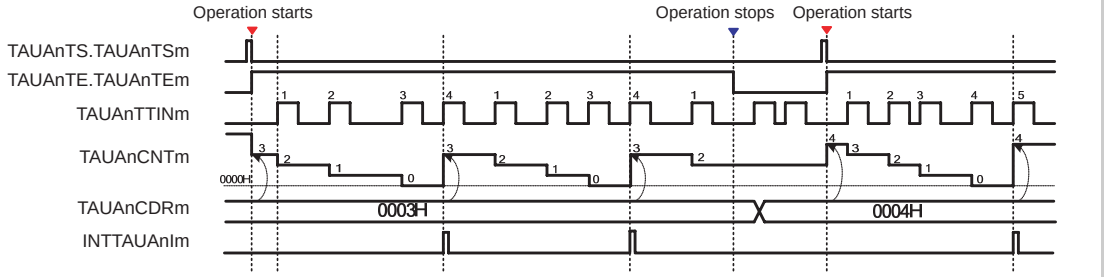


Figure 15-79 Operation stop and restart, TAUAnCMURm.TAUAnTIS[1:0] = 01_B

- The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1, which in turn sets TAUAnTE.TAUAnTEm to 0.
- TAUAnCNTm stops and the current value is retained. TAUAnTTINm continues and TAUAnCNTm ignores the valid edge.
- The counter can be restarted by setting TAUAnTS.TAUAnTSm to 1. TAUAnCNTm loads the TAUAnCDRm value, and then restarts count operation.

(c) Forced restart

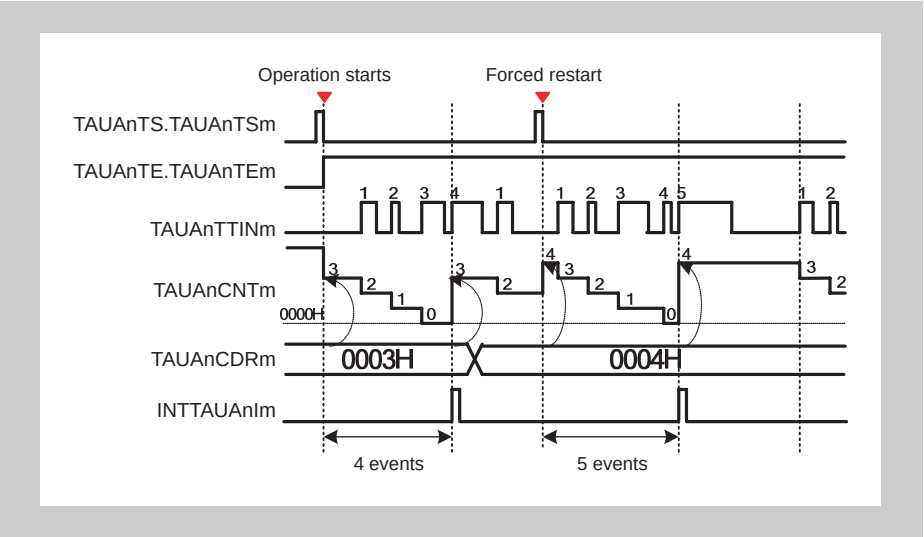


Figure 15-80 Forced restart, TAUAnCMURm.TAUAnTIS[1:0] = 01_B

A forced restart applies a change to TAUAnCDRm immediately.

- The counter can be restarted (without stopping it first), by setting TAUAnTS.TAUAnTSm to 1 during operation.
- The value of TAUAnCDRm is loaded to TAUAnCNTm and the counter awaits the next valid TAUAnTTINm input edge.

15.21.2 Clock divide function

(1) Overview

Summary This function is used as a frequency divider. The frequency of the input signal TAUAnTTINm is divided by a factor related to TAUAnCDRm, and the resulting signal is output to TAUAnTTOUTm.

Prerequisites

- TAUAnTTINm must have a fixed frequency.
- The operation mode must be set to interval timer mode. See *Table 15-97 "TAUAnCMORm settings for clock divide function" on page 759*.
- The channel output mode must be set to independent channel output mode 1. See *15.9 "Channel Output Modes" on page 627*.

Description The counter is started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. The current value of TAUAnCDRm is loaded to TAUAnCNTm and the counter starts to count down from this value, using TAUAnTTINm as the count clock.

When the counter value reaches 0000_H, INTTAUAnIm is generated and the TAUAnTTOUTm signal toggles. TAUAnCNTm loads the TAUAnCDRm value, and then subsequently continues operation.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the function starts to count down.

The counter can be stopped by setting TAUAnTT.TAUAnTTm = 1, which in turn sets TAUAnTE.TAUAnTEm = 0. TAUAnCNTm and TAUAnTTOUTm stop but retain their values. The function can be restarted by setting TAUAnTS.TAUAnTSm = 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUAnTS.TAUAnTSm = 1 during operation.

Conditions If the TAUAnCMORm.TAUAnMD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUAnTTOUTm does not toggle. This results in an inverted TAUAnTTOUTm signal compared to when TAUAnCMORm.TAUAnMD0 is set to 1. For details, see *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation When Counter Starts or Restarts" on page 639*.

Note The input TAUAnTTINm is sampled at the frequency of the operation clock, specified by TAUAnCMORm.TAUAnCKS[1:0] bits. As a result, the output cycle of TAUAnTTOUTm has an error of ± 1 operation clock cycle.

(2) Equations

- When rising edge detection is selected:
  TAUAnTTOUTm frequency = TAUAnTTINm frequency / [(TAUAnCDRm + 1) x 2]
- When falling edge detection is selected:
  TAUAnTTOUTm frequency = TAUAnTTINm frequency / [(TAUAnCDRm + 1) x 2]
- When rising and falling edge detection is selected:
  TAUAnTTOUTm frequency = TAUAnTTINm frequency / (TAUAnCDRm + 1)

(3) Block diagram and general timing diagram

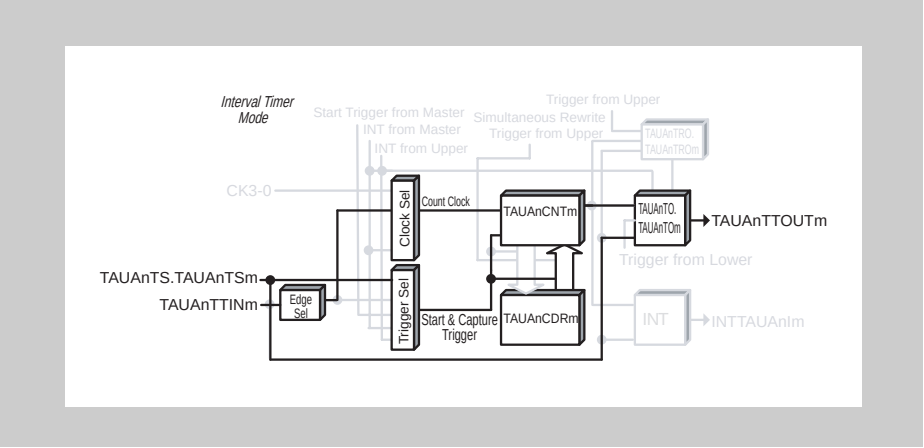


Figure 15-81 Block diagram for clock divide function

The following settings apply to the general timing diagram:

- INTTAUAnIm generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)
- Rising edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 01_B)

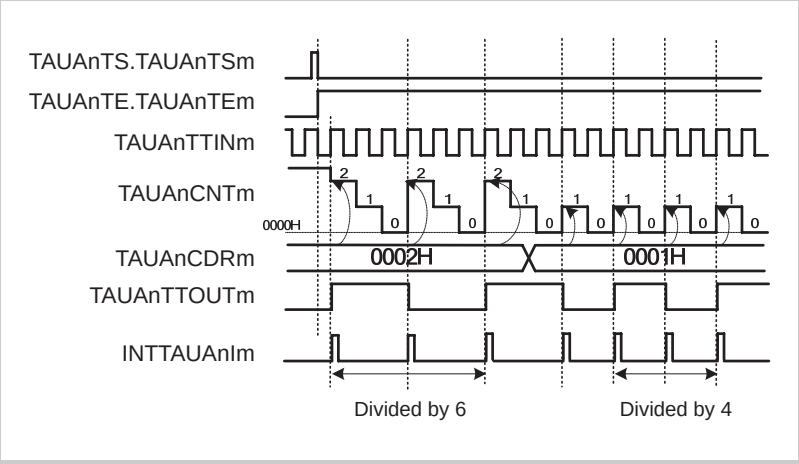


Figure 15-82 General timing diagram for clock divide function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAn STS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-97 TAUAnCMORm settings for clock divide function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	01: Valid TAUAnTTINm input edge is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-98 TAUAnCMURm settings for clock divide function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Table 15-99 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the clock divide function. Therefore, these registers must be set to 0.

Table 15-100 Simultaneous rewrite settings for clock divide function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(5) Operating procedure for clock divide function

Table 15-101 Operating procedure for clock divide function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-97 "TAUAnCMORm settings for clock divide function" on page 759* and *Table 15-98 "TAUAnCMURm settings for clock divide function" on page 759*. Set the value of the TAUAnCDRm register. Set the channel output mode by setting the control bits as described in *Table 15-99 "Control bit settings for independent channel output mode 1" on page 760*.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. When TAUAnCMORm.TAUAnMD0 is set to 1, INTTAUAnIm is generated and TAUAnTTOUTm toggles.
During operation	The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at all times.	When a TAUAnTTINm input edge is detected, TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm loads the TAUAnCDRm value, and then continues count operation. • INTTAUAnIm is generated. • TAUAnTTOUTm toggles. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnTTOUTm retain their current values.

Restart

(6) Specific timing diagrams

(a) TAUAnCDRm = 0000_H

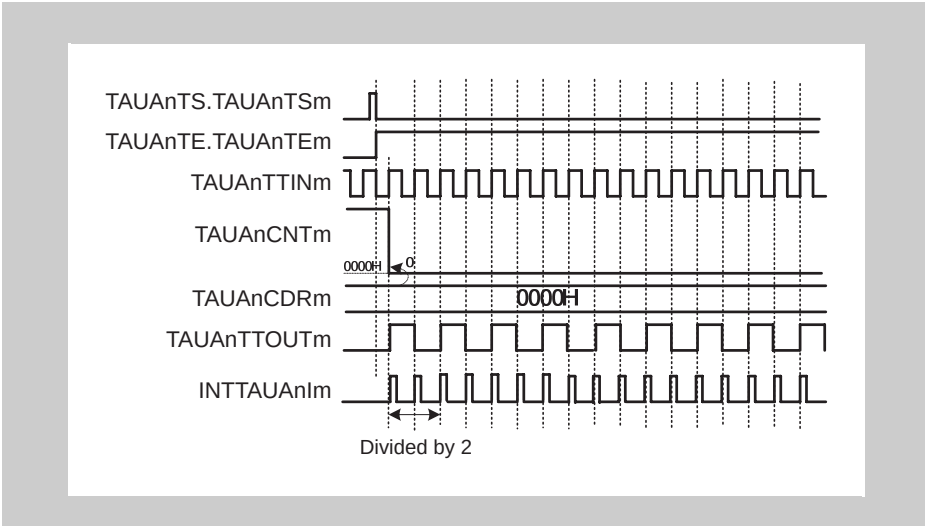


Figure 15-83 TAUAnCDRm = 0000_H, TAUAnCMORm.TAUAnMD0 = 1, TAUAnCMURm.TAUAnTIS[1:0] = 01_B

- If TAUAnCDRm is 0000_H, TAUAnCNTm is also always 0000_H.
- INTTAUAnIm is generated every count clock, resulting in TAUAnTTOUTm toggling every count clock.

Figure 15-83 "TAUAnCDRm = 0000_H, TAUAnCMORm.TAUAnMD0 = 1, TAUAnCMURm.TAUAnTIS[1:0] = 01_B" shows the approximate operation timing. In actuality, because there is delay due to the noise filter and synchronizer between the TAUAnIm pin and TAUAn, there is delay between TINm detection and TOUTm output.

(b) Restart

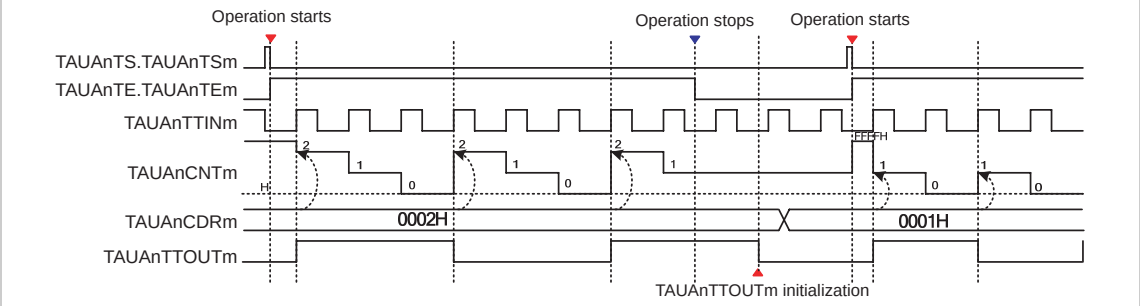


Figure 15-84 Restart, TAUAnCMORm.TAUAnMD0 = 1, TAUAnCMURm.TAUAnTIS[1:0] = 01_B

To reset the value of TAUAnTTOUTm:

- Set TAUAnTOE.TAUAnTOEm = 0 when the counter is stopped (TAUAnTE.TAUAnTEm = 0)
- Then write the either 0 or 1 to TAUAnTO.TAUAnTOm to set the new start value of TAUAnTTOUTm

15.21.3 TAUAnTTINm input position detection function

(1) Overview

Summary This function measures the duration between the function start and a TAUAnTTINm input signal.

Prerequisites
- The operation mode must be set to Count capture mode. See *Table 15-102 "TAUAnCMORm settings for TAUAnTTINm input position detection function" on page 765*.
- TAUAnTTOUTm is not used for this function.

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. The counter starts to count from 0000_H. When a valid TAUAnTTINm input stop edge is detected, the current TAUAnCNTm value is loaded to TAUAnCDRm and an interrupt (INTTAUAnIm) is generated. The counter continues counting.

When the counter reaches $FFFF_H$, the bit TAUAnCSRm.TAUAnOVF is set to 1 and the counter restarts from 0000_H. TAUAnCSRm.TAUAnOVF is cleared by setting TAUAnCSCm.CLOV.

Conditions If the TAUAnCMORm.TAUAnMD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details, see *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation When Counter Starts or Restarts" on page 639*.

(2) Equations

Function duration at a TAUAnTTINm input pulse =

count clock cycle × [($FFFF_H$+1 × TAUAnCSRm.TAUAnOVF) + (TAUAnCDRm capture value + 1)]

(3) Block diagram and general timing diagram

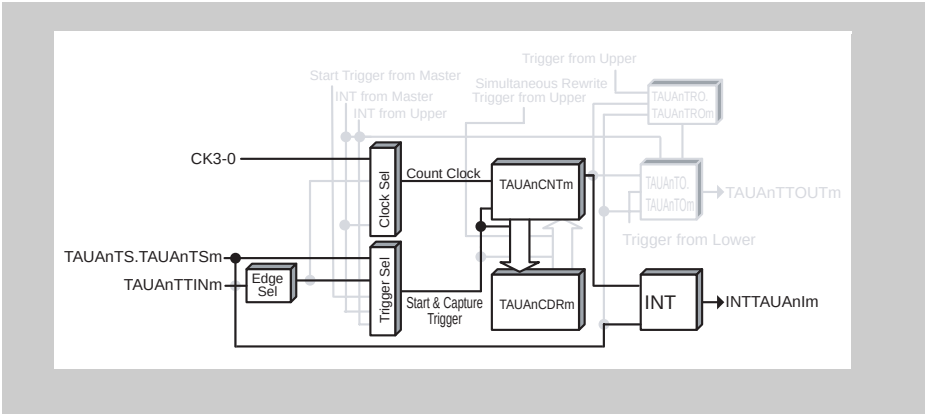


Figure 15-85 Block diagram for TAUAnTTINm input position detection function

The following settings apply to the general timing diagram:

- INTTAUAnIm not generated at operation start (TAUAnCMORm.TAUAnMD0 = 0)
- Falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)

<R>

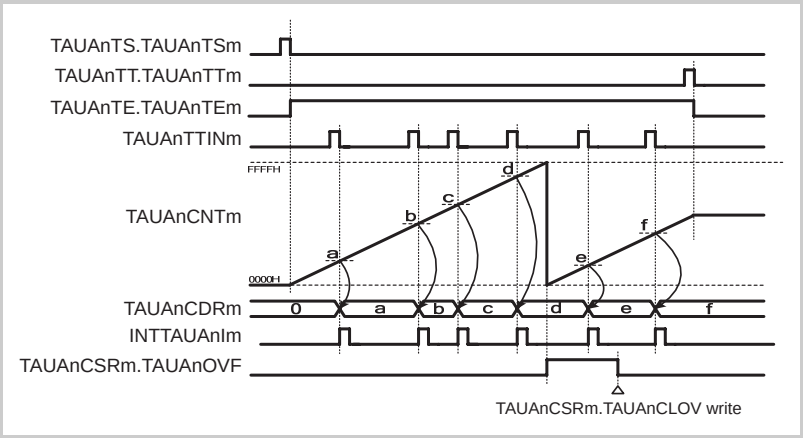


Figure 15-86 General timing diagram for TAUAnTTINm input position detection function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-102 TAUAnCMORm settings for TAUAnTTINm input position detection function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Not used, so set to 0
TAUAnSTS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the external capture trigger
TAUAnCOS[1:0]	01:Overflow (TAUAnCSRm.OVF) set upon counter overflow and cleared by a CPU instruction (by setting the TAUAnCSCm.TAUAnCLOV bit to 1).
TAUAnMD[4:1]	1011: Count capture mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start 1: Generates INTTAUAnIm at operation start

<R>

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-103 TAUAnCMURm settings for TAUAnTTINm input position detection function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm input position detection function. Therefore, these registers must be set to 0.

Table 15-104 Simultaneous rewrite settings for TAUAnTTINm Input Position Detection Function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(5) Operating procedure for TAUAnTTINm input position detection function

Table 15-105 Operating procedure for TAUAnTTINm input position detection function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-102 "TAUAnCMORm settings for TAUAnTTINm input position detection function" on page 765* and *Table 15-103 "TAUAnCMURm settings for TAUAnTTINm input position detection function" on page 765*. Set the value of the TAUAnCDRm register.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm to 1. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is set to 1 and the counter starts. INTTAUAnIm is generated when TAUAnCMORm.TAUAnMD0 is set to 1.
During operation	The TAUAnCMURm.TAUAnTIS[1:0] bits can be changed at any time. The TAUAnCDRm and TAUAnCSRm registers can be read at any time. The TAUAnCSC.CLOV bit can be set to 1.	TAUAnCNTm starts to count up from 0000_H. When a TAUAnTTINm valid edge is detected: • TAUAnCNTm transfers (captures) its value to TAUAnCDRm. • INTTAUAnIm is output. • The counter value is not cleared to 0000_H and TAUAnCNTm continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TAUAnTTm to 1. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

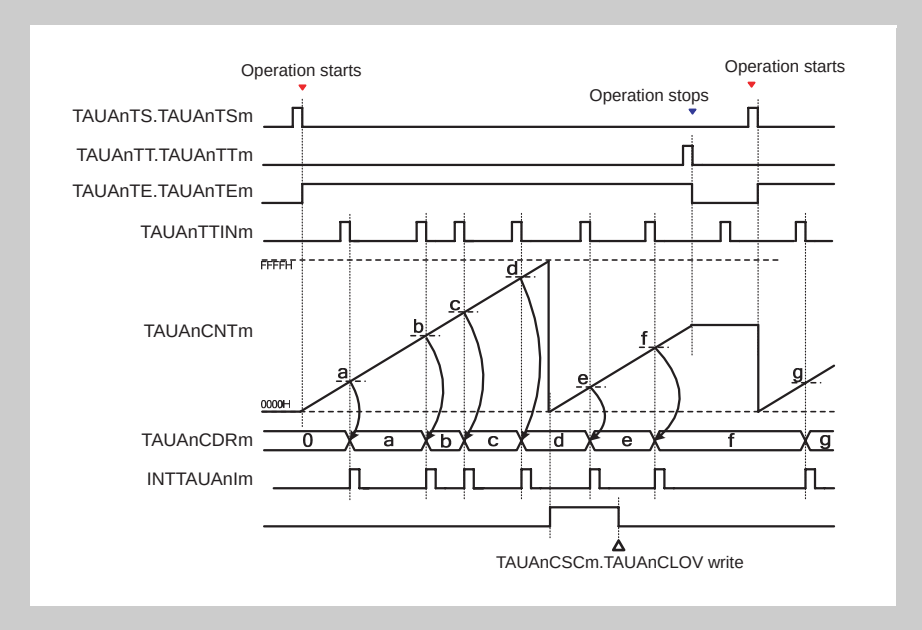


Figure 15-87 Operation stop and restart, TAUAnCMORm.TAUAnMD0 = 0, TAUAnCMURm.TAUAnTIS[1:0] = 00_B

- The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1, which in turn sets TAUAnTE.TAUAnTEm to 0.
- TAUAnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUAnTTINm input edges are ignored.
- The counter can be restarted by setting TAUAnTS.TAUAnTSm to 1. TAUAnCNTm restarts to count from 0000_H.

15.22 Synchronous Channel Operation Functions

This section lists all the synchronous channel operation functions provided by TAUA. For a general overview of synchronous channel operation, see *15.4 “Functional Description” on page 608*.

15.23 Synchronous PWM Signal Functions Triggered at Regular Intervals

This chapter describes functions that generate PWM signals at regular intervals.

- *15.23.1 “PWM output function”*
- *15.23.2 “Trigger start PWM output function”*
- *15.23.3 “Delay pulse output function”*
- *15.23.4 “AD conversion trigger output function type 1”*

15.23.1 PWM output function

(1) Overview

Summary This function generates multiple PWM outputs by using a master and multiple slave channels. It enables the pulse cycle (frequency) and the duty cycle of the TAUAnTTOUTm to be set. The pulse cycle is set in the master channel. The duty cycle is set in the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to interval timer mode. See *Table 15-106 “TAUAnCMORm settings for the master channel of the PWM output function” on page 773*.
- The operation mode of the slave channel(s) must be set to one-count mode. See *Table 15-109 “TAUAnCMORm settings for the slave channel of the PWM output function” on page 775*.
- TAUAnTTOUTm is not used for the master channel of this function.
- The channel output mode of the slave channel(s) must be set to synchronous channel output mode 1 (*15.9 “Channel Output Modes” on page 627*).

Description The counters are started by setting the channel trigger bits (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm = 1, enabling count operation. The current value of TAUAnCDRm is loaded to TAUAnCNTm and the counters start to count down from these values. INTTAUAnIm is generated on the master channel, and PWM output is achieved by setting and resetting TAUAnTTOUTm (slave).

- Master channel:

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed and INTTAUAnIm is generated. TAUAnCNTm loads the TAUAnCDRm value, and then counts down.

- Slave channel(s)

  The INTTAUAnIm of the master channel triggers the counter of the slave channel(s). The current value of TAUAnCDRm (slave) is loaded to TAUAnCNTm (slave) and the counter starts to count down from this value. The TAUAnTTOUTm signal becomes active.

  When the counter reaches 0000_H, i.e. duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal becomes inactive. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel, and thus the start of the next pulse cycle.

The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channel(s), which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel(s) stop but retain their values. The counters can be restarted by setting TAUAnTS.TAUAnTSm to 1.

Conditions Simultaneous rewrite can be used with this function. See *15.8 “Simultaneous Rewrite” on page 615*.

(2) Equations

Pulse cycle = (TAUAnCDRm (master) + 1) x count clock cycle

Duty cycle [%] = (TAUAnCDRm (slave) / (TAUAnCDRm (master) + 1)) x 100

– Duty cycle = 0 %

TAUAnCDRm (slave) = 0000_H

– Duty cycle = 100 %

TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

(3) Block diagram and general timing diagram

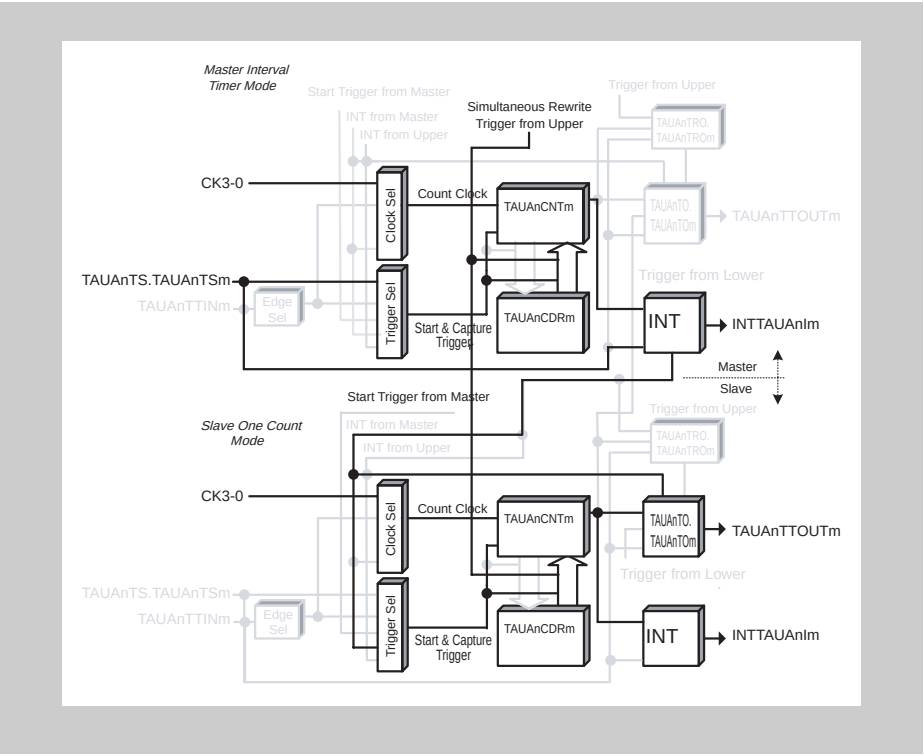


Figure 15-88 Block diagram for PWM output function

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUAnTOL.TAUAnTOLm = 0)

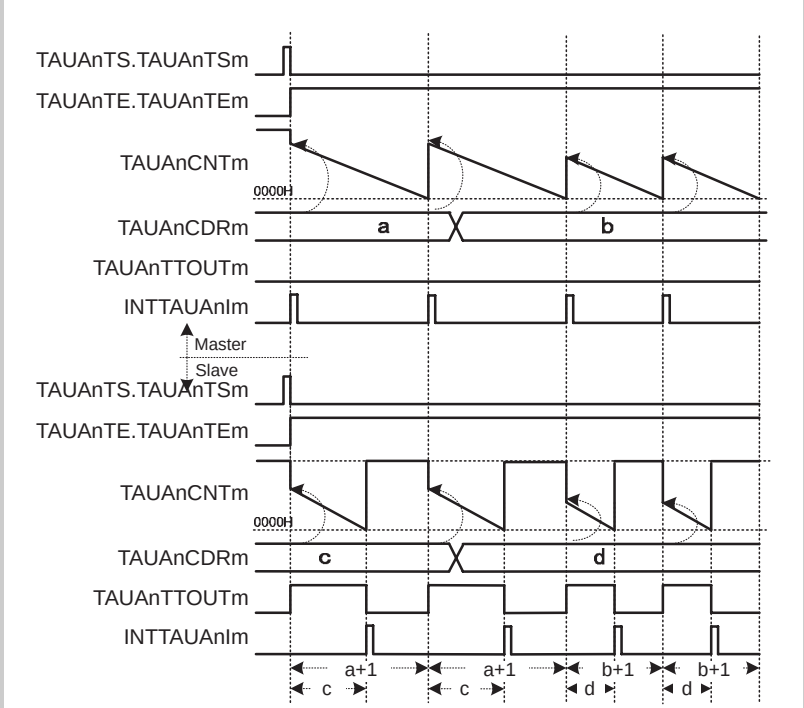


Figure 15-89 General timing diagram for PWM output function

Note The interval between the slave channel starting to count and an interrupt being generated is the value of corresponding TAUAnCDRm, whereas for the master channel the interval is the corresponding TAUAnCDRm + 1.

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-106 TAUAnCMORm settings for the master channel of the PWM output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-107 TAUAnCMURm settings for the master channel of the PWM output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-108 Simultaneous rewrite settings for the master channel of the PWM output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

Note If the TAUAnRDS.TAUAnRDSm bit is 1, there must be a channel higher than the master channel that operates with simultaneous rewrite trigger output function type 1.

- Set up the operation as follows:

  Channel setting for simultaneous rewrite trigger output function type 1: TAUAnRDCm = 1, TAUAnRDS = 1

  Note that the TAUAnCDR setting for this channel is as follows:

  = ((TAUAnCDR setting of master channel subject to simultaneous rewriting +1) × number of interrupts) - 1

- Master channel: TAUAnRDCm = 0, TAUAnRDS = 1
- Slave channel: TAUAnRDCm = 0, TAUAnRDS = 1

When the CDRn (slave) setting is greater than the CDRn (master) setting + 1, the duty ratio exceeds 100%, but 100% output is assumed.

(5) Register settings for the slave channel(s)

(a) TAUAnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-109 TAUAnCMORm settings for the slave channel of the PWM output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Enables the start trigger during operation

(b) TAUAnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-110 TAUAnCMURm settings for the slave channel of the PWM output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the slave channel(s)

Table 15-111 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel operation
TAUAnTOC.TAUAnTOCm	0: Operation mode 1
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-112 Simultaneous rewrite settings for the slave channel of the PWM output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for PWM output function

Table 15-113 Operating procedure for PWM output function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 773*. Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for the slave channel(s)" on page 775*. Set the values of the TAUAnCDRm registers of all channels.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUAnIm is generated on the master channel and TAUAnTTOUTm (slave) becomes active.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	TAUAnCNTm of the master channel loads TAUAnCDRm, and then counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated. • TAUAnCNTm (master) loads the TAUAnCDRm value, and then continues count operation. • TAUAnCNTm (slave) loads the TAUAnCDRm value, and then counts down. • TAUAnTTOUTm (slave) becomes active. When TAUAnCNTm (slave) reaches 0000_H: • INTTAUAnIm (slave) is generated. • TAUAnTTOUTm (slave) becomes inactive.
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0%

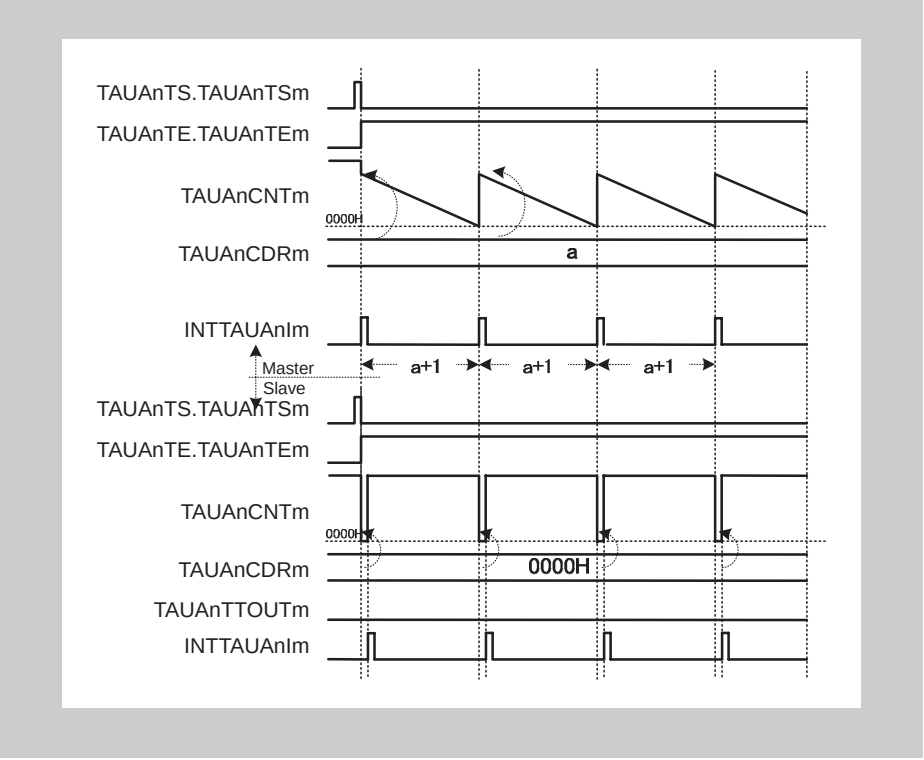


Figure 15-90 TAUAnCDRm (slave) = 0000_H, positive logic (TAUAnTOL.TAUAnTOLm (slave) = 0)

- Every time the master channel generates an interrupt (INTTAUAnIm), 0000_H is loaded to TAUAnCNTm (slave). Therefore, TAUAnCNTm (slave) cannot start to count and TAUAnTTOUTm remains at not active state.
- TAUAnCNTm (slave) loads the TAUAnCDRm value and an interrupt is generated.

(b) Duty cycle = 100%

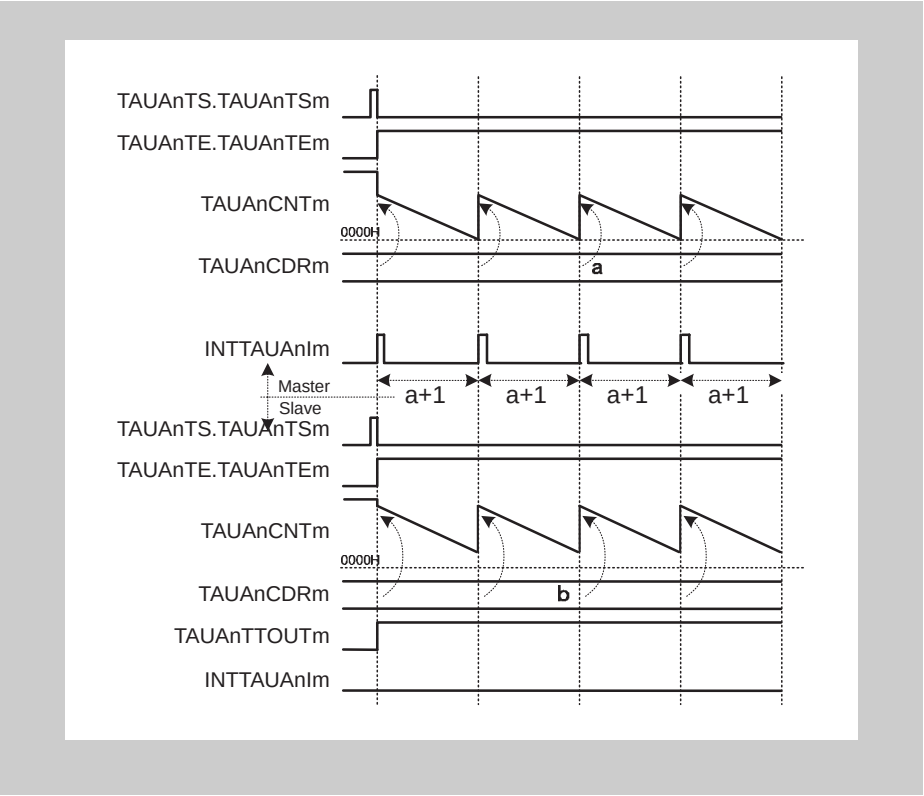


Figure 15-91 TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1, positive logic (TAUAnTOL.TAUAnTOLm (slave) = 0)

- If the value TAUAnCDRm (slave) is higher than the value TAUAnCDRm (master), the counter of the slave channel cannot reach 0000_H and interrupts are not generated. The TAUAnTTOUTm remains at active state.

(c) Stop and restart operation

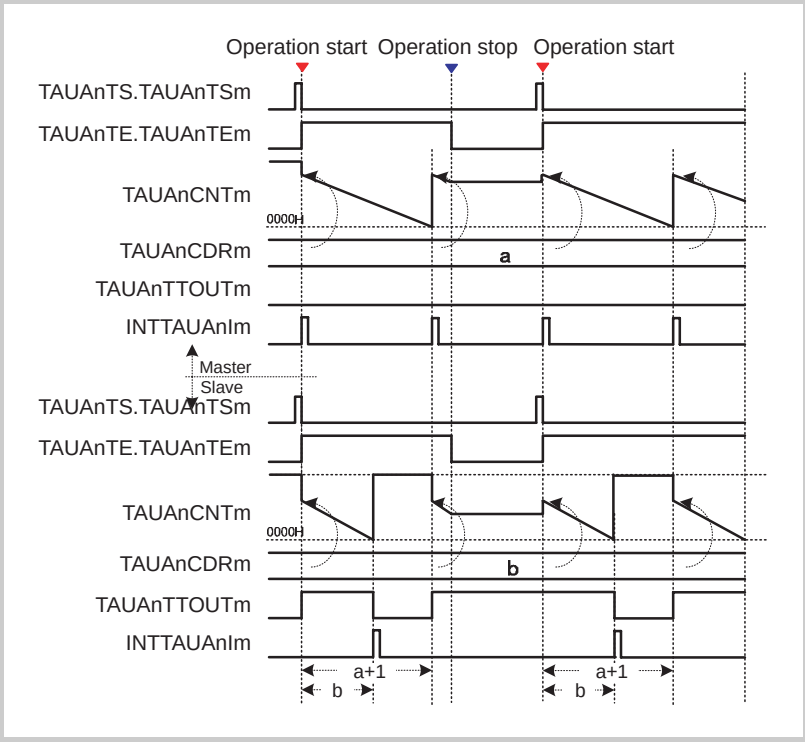


Figure 15-92 Stop and restart operation, positive logic (TAUAnTOL.TAUAnTOLm (slave) = 0)

- The counter can be stopped by setting TAUAnTT.TAUAnTTm of the master and slave channel(s) to 1, which in turn sets TAUAnTE.TAUAnTEm to 0.
- TAUAnCNTm and TAUAnTTOUTm of all channels stop and the current values are retained. No interrupts are generated.
- The counter can be restarted by setting TAUAnTS.TAUAnTSm of master and slave channel(s) to 1. TAUAnCNTm loads the TAUAnCDRm value of master and slave channels, and then starts counting down from this value.

15.23.2 Trigger start PWM output function

(1) Overview

Summary This function generates a PWM output using a master and a slave channel. It enables the pulse cycle (frequency) and the duty cycle of the TAUAnTTOUTm to be set. The duty cycle is specified using the master channel. The pulse width is specified using the slave channel. The Trigger Start PWM output function is identical to PWM output function except that the master channel of this function can be reset by a valid TAUAnTTINm input edge.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to interval timer mode. See *Table 15-114 "TAUAnCMORm settings for the master channel of the trigger start PWM output function" on page 784*.
- The operation mode of the slave channel must be set to one-count mode. See *Table 15-117 "TAUAnCMORm settings for the slave channel of the trigger start PWM output function" on page 786*.
- The channel output mode of the slave channel must be set to synchronous channel output mode 1. See *15.9 "Channel Output Modes" on page 627*.
- TAUAnTTOUTm is not used for the master channel of this function.

Description The counters (master and slave) are started by setting the channel trigger bits (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation.

TAUAnCNTm loads the current value of TAUAnCDRm, and then starts counting down from this value. INTTAUAnIm is generated on the master channel, and PWM output is achieved by setting and resetting TAUAnTTOUTm (slave).

- Master channel:

  The current value of TAUAnCDRm is loaded to the counter (TAUAnCNTm), INTTAUAnIm is generated and the counter starts to count down from this value.

  When the counter reaches 0000_H, pulse cycle time has elapsed, INTTAUAnIm is generated and the TAUAnCNTm (master and slave) load the current TAUAnCDRm values.

  If a valid TAUAnTTINm input edge is detected, the counter of the master channel loads the current TAUAnCDRm value, restarts counting down, and then generates an interrupt.

- Slave channel:

  When the slave detects an interrupt from the master channel, it starts to count down from the current value of TAUAnCDRm. The TAUAnTTOUTm signal becomes active.

  When the counter reaches 0000_H, duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel.

The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channel, which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel stop but retain their values. The counters can be restarted by setting TAUAnTS.TAUAnTSm to 1.

Conditions Simultaneous rewrite can be used with this function. See *15.8 "Simultaneous Rewrite" on page 615*.

(2) Equations

Pulse cycle = (TAUAnCDRm (master) + 1) x count clock cycle

Duty cycle [%] = [TAUAnCDRm (slave) / (TAUAnCDRm (master) + 1)] x 100

– Duty cycle = 0 %

TAUAnCDRm (slave) = 0000_H

– Duty cycle = 100 %

TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

(3) Block diagram and general timing diagram

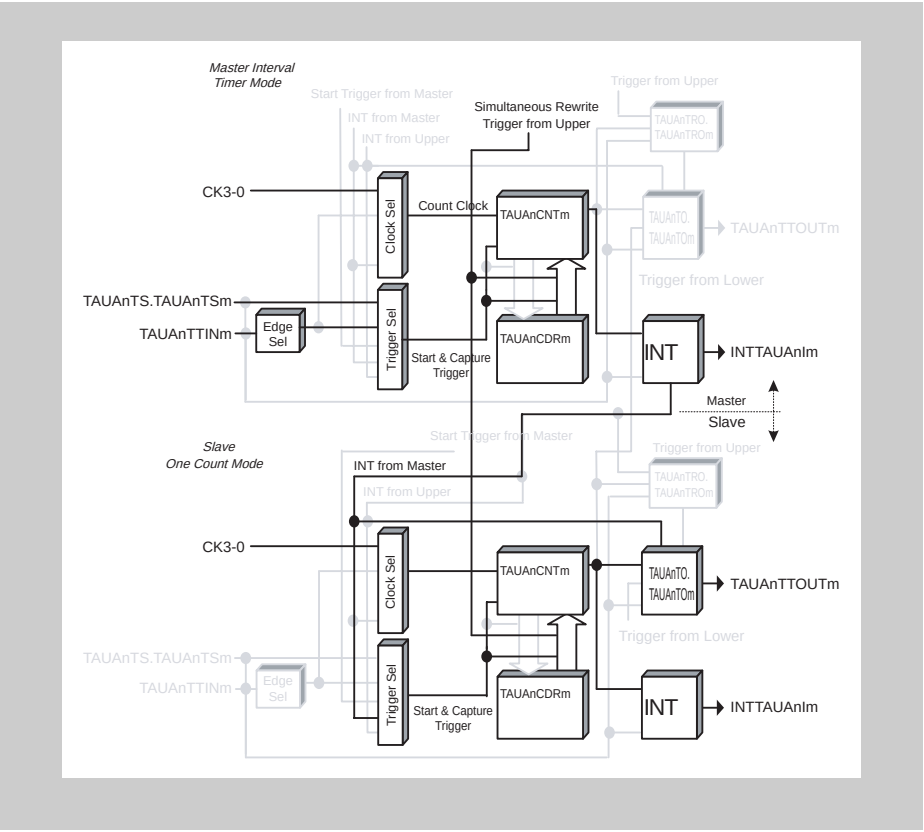


Figure 15-93 Block diagram for trigger start PWM output function

The following settings apply to the general timing diagram:

- Rising edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 01_B)
- Positive logic (TAUAnTOL.TAUAnTOLm (slave) = 0)

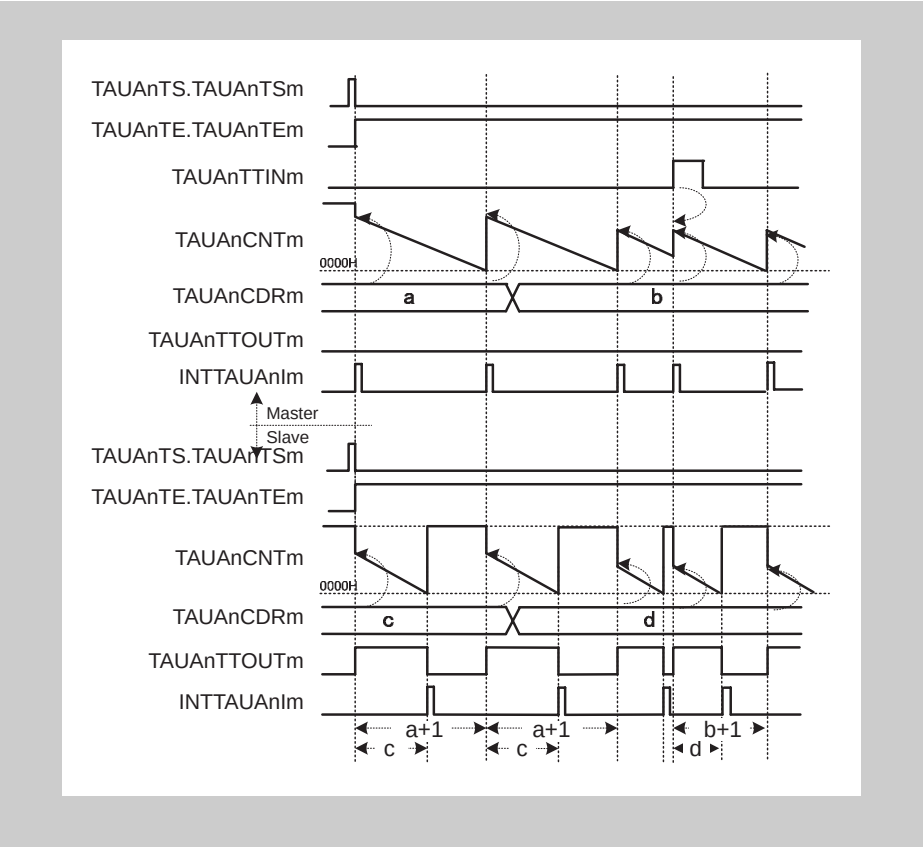


Figure 15-94 General timing diagram for trigger start PWM output function

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-114 TAUAnCMORm settings for the master channel of the trigger start PWM output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-115 TAUAnCMURm settings for the master channel of the trigger start PWM output function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-116 Simultaneous rewrite settings for the master channel of the trigger start PWM output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

Note If the TAUAnRDS.TAUAnRDSm bit is 1, there must be a channel higher than the master channel that operates with simultaneous rewrite trigger output function type 2.

- Channel setting for simultaneous rewrite trigger output function type 2: TAUAnRDCm = 1, TAUAnRDS = 1
- Master channel: TAUAnRDCm = 0, TAUAnRDS = 1
- Slave channel: TAUAnRDCm = 0, TAUAnRDS = 1

(5) Register settings for the slave channel

(a) TAUAnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-117 TAUAnCMORm settings for the slave channel of the trigger start PWM output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Enables the start trigger during operation The value of the TAUAnMD0 bit of the master and slave channel must be identical.

<R>

(b) TAUAnCMURm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-118 TAUAnCMURm settings for the slave channel of the trigger start PWM output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the slave channel

Table 15-119 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel operation
TAUAnTOC.TAUAnTOCm	0: Operation mode 1
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-120 Simultaneous rewrite settings for the slave channel of the trigger start PWM output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for trigger start PWM output function

Table 15-121 Operating procedure for trigger start PWM output function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 784*. Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for the slave channel" on page 786*. Set the values of the TAUAnCDRm registers of all channels.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUAnIm is generated on the master channel.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	TAUAnCNTm of the master channel loads TAUAnCDRm, and then counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated. • TAUAnCNTm (master) loads the TAUAnCDRm value, and then continues count operation. • TAUAnCNTm (slave) loads the TAUAnCDRm value, and then starts counting down. • TAUAnTTOUTm (slave) becomes active. When TAUAnCNTm of the slave = 0000_H: • INTTAUAnIm (slave) is generated. • TAUAnTTOUTm (slave) becomes inactive. If a TAUAnTTINm input is detected on the master channel while the TAUAnCNTm of the master channel is counting down: • TAUAnCNTm (master and slave) loads the TAUAnCDRm value, and then counts down. • INTTAUAnIm (master) is generated. • TAUAnTTOUTm (slave) becomes active.
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0%

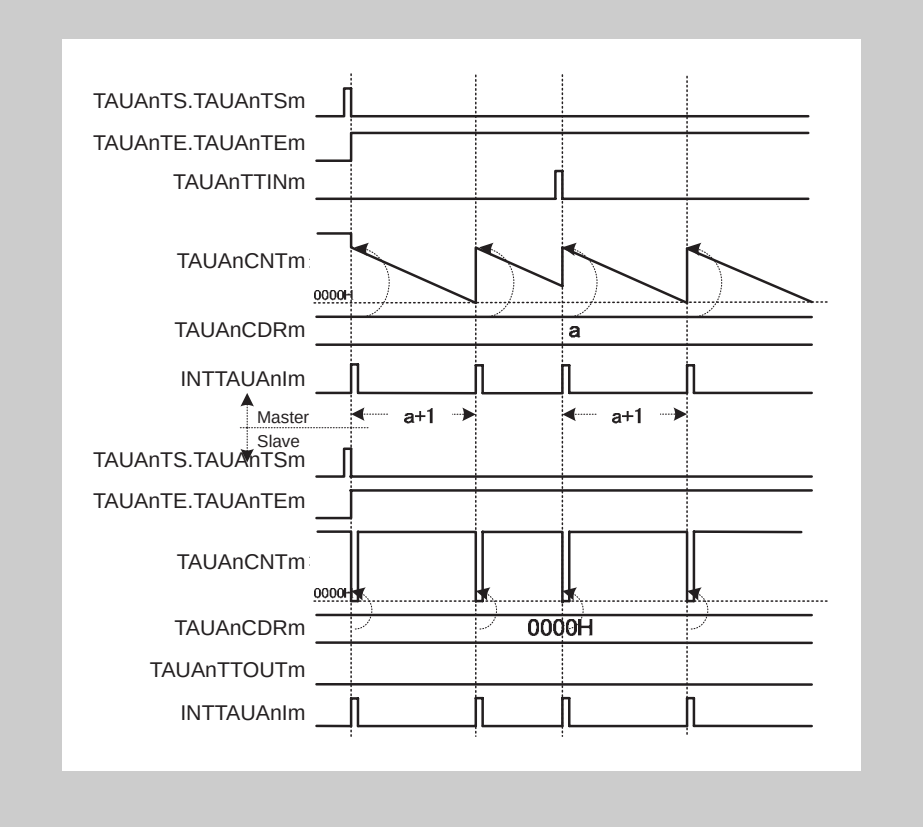


Figure 15-95 TAUAnCDRm (slave) = 0000_H, positive logic (TAUAnTOL.TAUAnTOLm (slave) = 0) falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)

- Every time the master channel generates an interrupt (INTTAUAnIm), 0000_H is loaded to TAUAnCNTm (slave). Therefore, TAUAnCNTm (slave) cannot start to count and TAUAnTTOUTm remains at not active state.
- TAUAnCNTm (slave) generates an interrupt every time the value of TAUAnCDRm is loaded.

  The detection of a valid TAUAnTTINm input edge has no effect on TAUAnTTOUTm (slave).

(b) Duty cycle = 100%

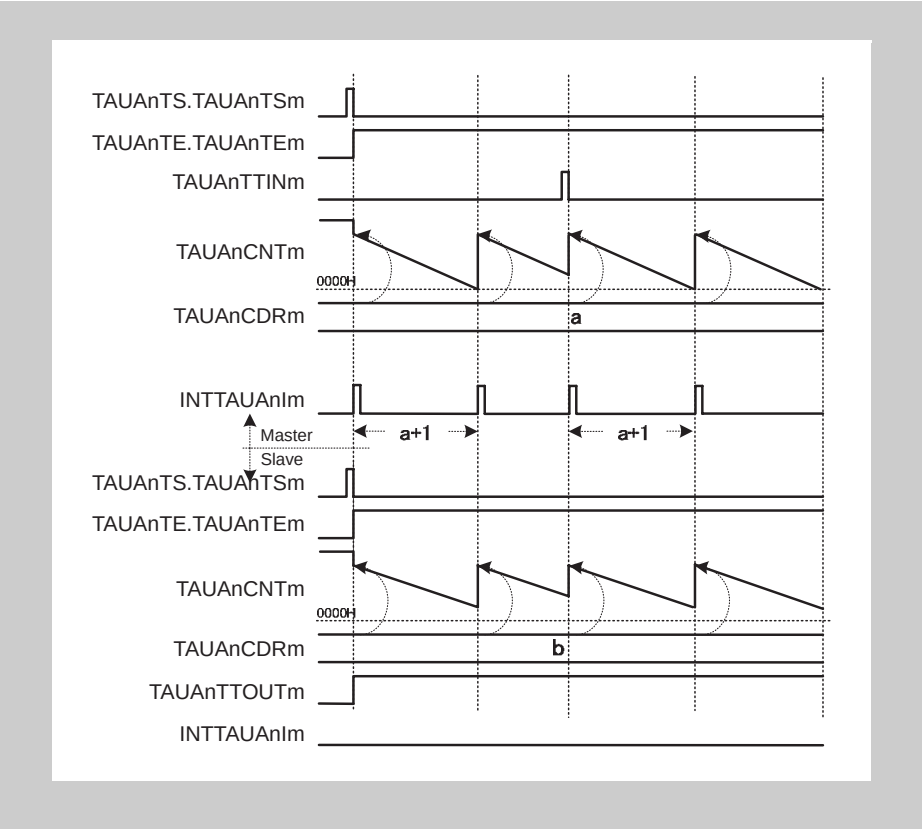


Figure 15-96 TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1, positive logic (TAUAnTOL.TAUAnTOLm (slave) = 0) falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)

- If the value TAUAnCDRm (slave) is higher than the value TAUAnCDRm (master), the counter of the slave channel cannot reach 0000_H and no interrupt is generated.

  The TAUAnTTOUTm remains at active state.

  The detection of a valid TAUAnTTINm input edge has no effect on TAUAnTTOUTm (slave).

(c) TAUAnTTINm detection and active slave counter

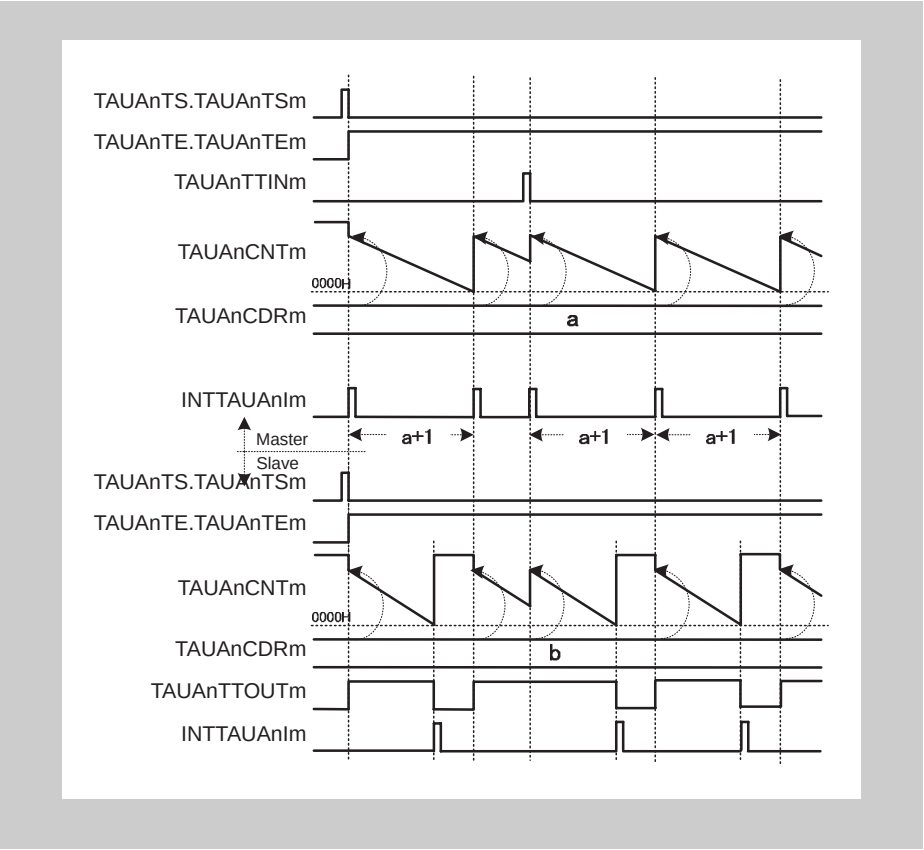


Figure 15-97 Positive logic (TAUAnTOL.TAUAnTOLm (slave) = 0), falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)

- If TAUAnCNTm (slave) loads the value of TAUAnCDRm (slave) while it is counting down, TAUAnTTOUTm does not change and extends the duty cycle.

  The duty cycle does not correspond to the value of the data register of the slave channel.

15.23.3 Delay pulse output function

(1) Overview

Summary This function outputs two signals. The reference signal has a defined pulse width and pulse cycle specified using the master channel and slave channel 1. Slave channels 2 and 3 output the reference signal with a specified delay. The delay signal is identical to the reference signal, but delayed by amount specified in slave channel 2.

The signal values are specified in the following way:

- The pulse cycle is specified using the master channel.
- The duty cycle of the reference signal is specified using slave channel 1. The duty cycle of the delay signal is specified using slave channel 3.
- The delay is specified in slave channel 2.

Prerequisites

- Four channels
- The operation mode of the master channel must be set to interval timer mode. See *Table 15-122 “TAUAnCMORm settings for the master channel of the delay pulse output function” on page 796*.
- The operation mode of slave channel 1 and 2 must be set to one-count mode. See *Table 15-125 “TAUAnCMORm settings for slave channel 1 of the delay pulse output function” on page 798*.
- The operation mode of slave channel 3 must be set to pulse one-count mode. See *Table 15-129 “TAUAnCMORm settings for slave channel 2 of the delay pulse output function” on page 800*.
- TAUAnTTOUTm is not used for the master channel and slave channel 2.
- The channel output mode of slave channel 1 must be set to synchronous channel output mode 1 (see *15.9 “Channel Output Modes” on page 627*).
- The channel output mode of slave channel 3 must be set to independent channel output mode 2 (see *15.9 “Channel Output Modes” on page 627*).

Description The counters of the channel group are started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation.

- Master channel:

  The current value of TAUAnCDRm is loaded to TAUAnCNTm and the counter starts to count down from this value. INTTAUAnIm is generated on the master channel.
  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed and INTTAUAnIm is generated. The counter reloads the TAUAnCDRm value, and then counts down.

- Slave channels 1 and 2:

  When the slave channels 1 and 2 detect an interrupt from the master channel, they start to count down from the current value of TAUAnCDRm. The TAUAnTTOUTm signal (slave 1) becomes active.

  – Slave channel 1:

    When the counter of slave channel 1 reaches 0000_H, duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel.

– Slave channel 2:

When the counter of slave channel 2 reaches 0000_H, delay time has elapsed and INTTAUAnIm is generated. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel.

INTTAUAnIm (slave 2) triggers the counter of slave channel 3

- Slave channel 3:

When slave channel 3 detects an interrupt from slave channel 2, it starts to count down from the current value of TAUAnCDRm. INTTAUAnIm is generated and the TAUAnTTOUTm signal (slave 3) is set.

When the counter of slave channel 3 reaches 0000_H, duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset.

The output from slave channel 3 is the delayed PWM pulse

The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TAUAnTSm to 1.

Conditions Simultaneous rewrite can be used with this function. See *15.8 "Simultaneous Rewrite" on page 615*.

Equations Pulse cycle = (TAUAnCDRm (master) + 1) × count clock cycle

Duty width 1 = (TAUAnCDRm (slave 1)) × count clock cycle

Delay width = (TAUAnCDRm (slave 2) + 1) × count clock cycle

Duty width 2 = (TAUAnCDRm (slave 3)) × count clock cycle

However, the duty width setting is assumed to be in the following range:

0000H ≤ TAUAnCDRm (slave 2) < TAUAnCDRm (master)

Notes

1. The output waveform of TAUAnTOUTm (slave 3) is the output waveform of TAUAnTOUTm (slave 1) delayed by the amount of delay generated by slave 2. It is not possible to delay the waveform by a pulse cycle or more.
2. If the TAUAnINTm signal for slave 2 is generated during slave 3 counting, slave 3 resumes operation. Therefore, the output waveform of TAUAnTOUTm (slave 3) stays at the active level. (In this case, it is not possible for TOUTn (slave channel 3) to output a waveform that delays the TOUTn (slave channel 1) reference pulse.)

(2) Block diagram and general timing diagram

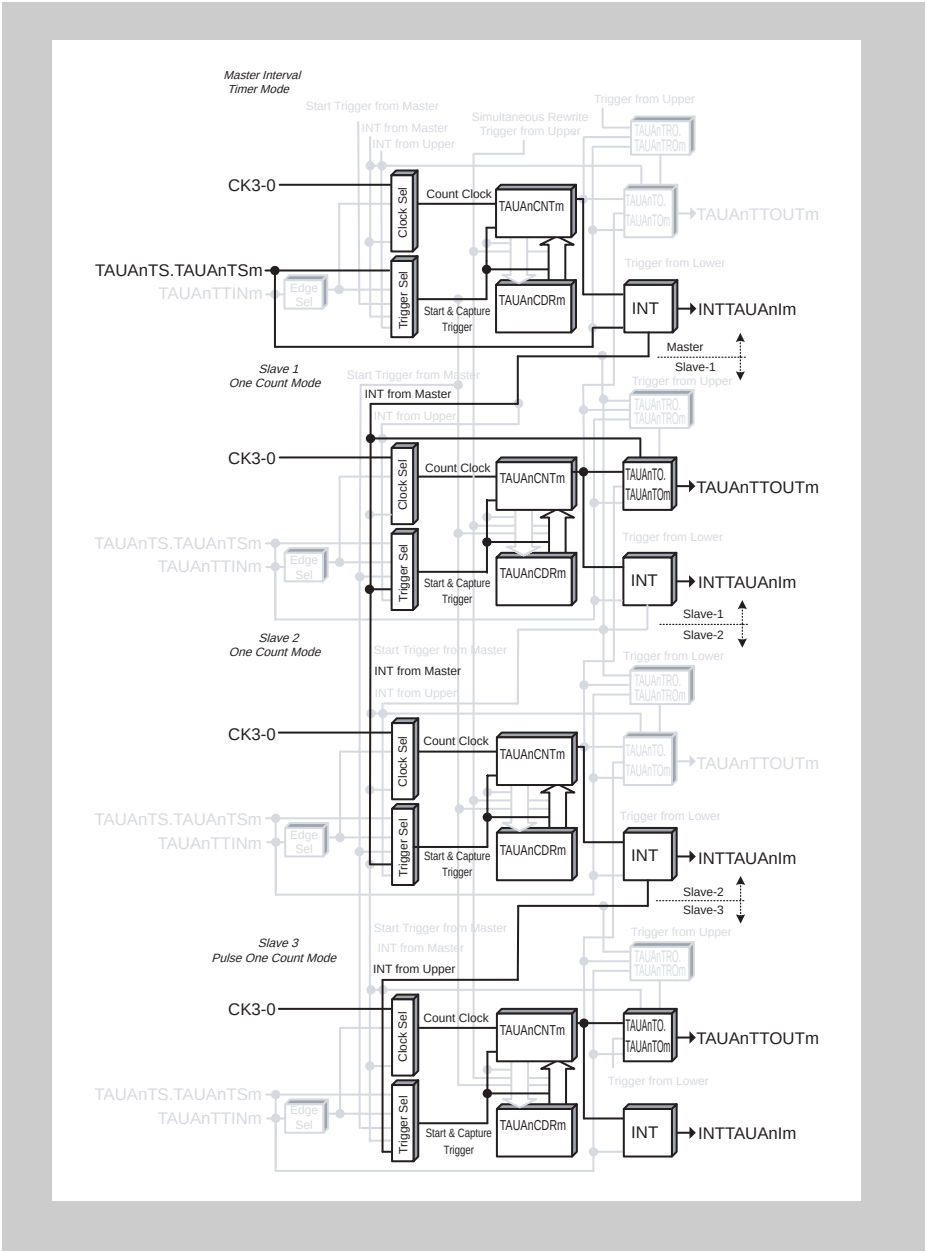


Figure 15-98 Block diagram for delay pulse output function

The following settings apply to the general timing diagram:

- All channels
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)

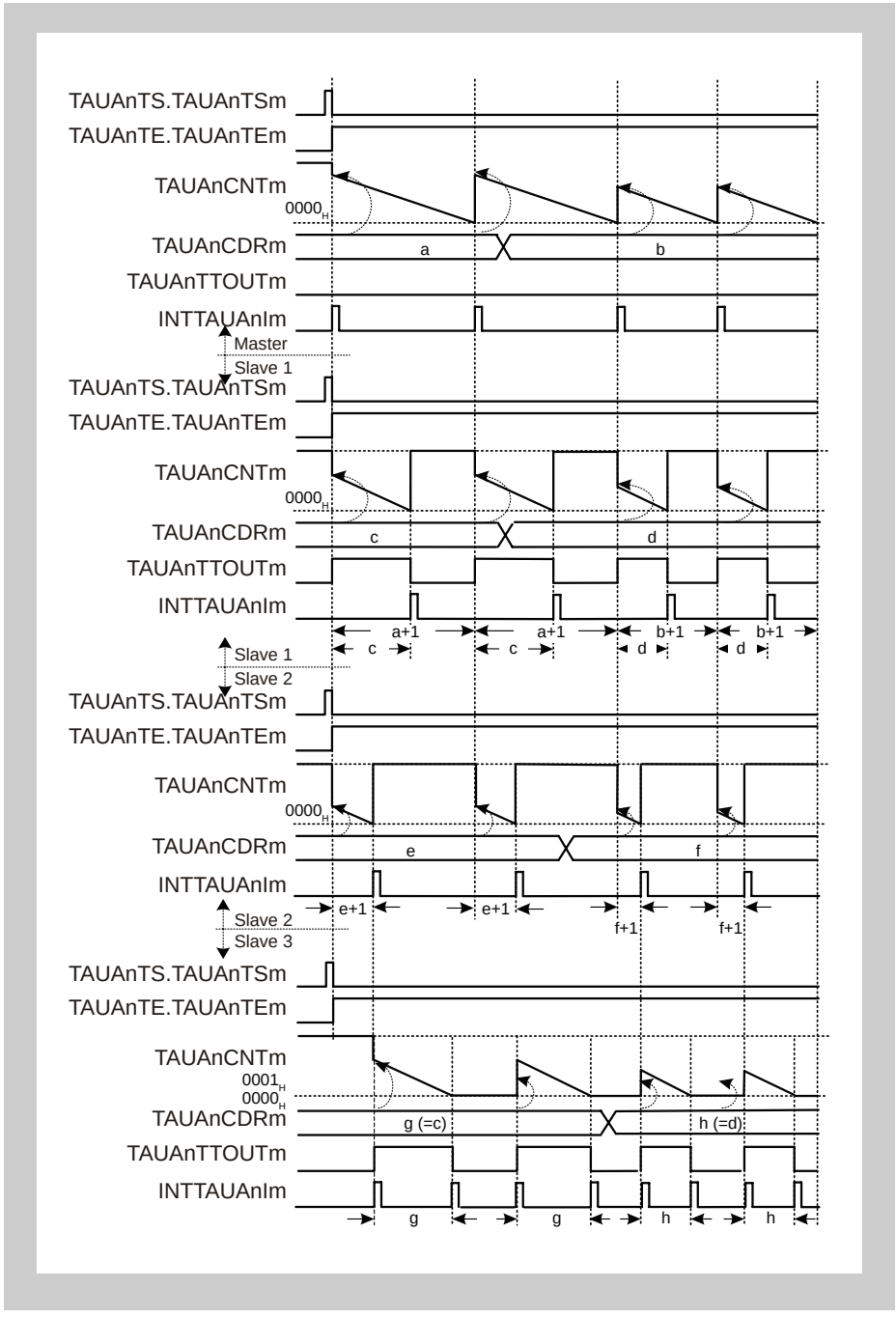


Figure 15-99 General timing diagram for delay pulse output function

(3) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAn MD[4:1]				TAUAn MD0

Table 15-122 TAUAnCMORm settings for the master channel of the delay pulse output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-123 TAUAnCMURm settings for the master channel of the delay pulse output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the master channel

Because the channel output mode is not used by the master channel of this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-124 Simultaneous rewrite settings for the master channel of the delay pulse output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(4) Register settings for slave channel 1

(a) TAUAnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-125 TAUAnCMORm settings for slave channel 1 of the delay pulse output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Enables the start trigger during operation

(b) TAUAnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-126 TAUAnCMURm settings for slave channel 1 of the delay pulse output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for slave channel 1

Table 15-127 Control bit settings for slave channel 1 of the synchronous channel output mode 1

<R>

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel operation
TAUAnTOC.TAUAnTOCm	Operation mode 1
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-128 Simultaneous rewrite settings for slave channel 1 of the delay pulse output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 2

(a) TAUAnCMORm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAn STS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-129 TAUAnCMORm settings for slave channel 2 of the delay pulse output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Enables the start trigger during operation

(b) TAUAnCMURm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-130 TAUAnCMURm settings for slave channel 2 of the delay pulse output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for slave channel 2

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for slave channel 2

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-131 Simultaneous rewrite settings for slave channel 2 of the delay pulse output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channel 3

(a) TAUAnCMORm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-132 TAUAnCMORm settings for slave channel 3 of the delay pulse output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	101: INTTAUAnIm of the upper channel (m-1) is the start trigger, regardless of the master setting
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	1010: Pulse one-count mode
TAUAnMD0	1: Enables the start trigger during operation

(b) TAUAnCMURm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-133 TAUAnCMURm settings for slave channel 3 of the delay pulse output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for slave channel 3

Table 15-134 Control bit settings for independent channel output mode 2

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for slave channel 3

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-135 Simultaneous rewrite settings for slave channel 3 of the delay pulse output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not used to detect an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is used to detect the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for delay pulse output function

Table 15-136 Operating procedure for delay pulse output function (1/2)

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(3) "Register settings for the master channel" on page 796*. Slave channel 1: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for slave channel 1" on page 798*. Slave channel 2: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for slave channel 2" on page 800*. Slave channel 3: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(6) "Register settings for slave channel 3" on page 802*. Set the values of the TAUAnCDRm registers of all channels.	Channel operation is stopped.

Table 15-136 Operating procedure for delay pulse output function (2/2)

	Operation	Status of TAUAn
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm (master and slave channels) is set to 1 and the counters of the master channel and slave channels 1 and 2 start. INTTAUAnIm is generated on the master channel and TAUAnTTOUTm (slave 1) is set.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	TAUAnCNTm loads the TAUAnCDRm value of the master channel and slave channels 1 and 2, and then counts down. When the counter of the master channel reaches 0000_H: • INTTAUAnIm (master) is generated. • TAUAnCNTm (master) reloads the TAUAnCDRm value, and then continues count operation. • TAUAnCNTm (slave 1 and slave 2) reload the TAUAnCDRm value and start counting down. • TAUAnTTOUTm (slave 1) is set. When TAUAnCNTm (slave 1) reaches 0000_H: • INTTAUAnIm (slave 1) is generated. • TAUAnTTOUTm (slave 1) is reset. When TAUAnCNTm (slave 2) reaches 0000_H: • INTTAUAnIm (slave 2) is generated. • TAUAnTTOUTm (slave 3) is set. • TAUAnCNTm (slave 3) reloads the TAUAnCDRm value, and then starts counting down. When TAUAnCNTm (slave 3) reaches 0000_H: • INTTAUAnIm (slave 3) is generated. • TAUAnTTOUTm (slave 3) is reset.
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

15.23.4 AD conversion trigger output function type 1

(1) Overview

Summary This function is identical to *15.23.1 "PWM output function" on page 770*except that TAUAnTTOUTm is not output.

This is achieved by setting the channel output mode of the slave to independent channel output mode controlled by software.

(2) Block diagram and general timing diagram

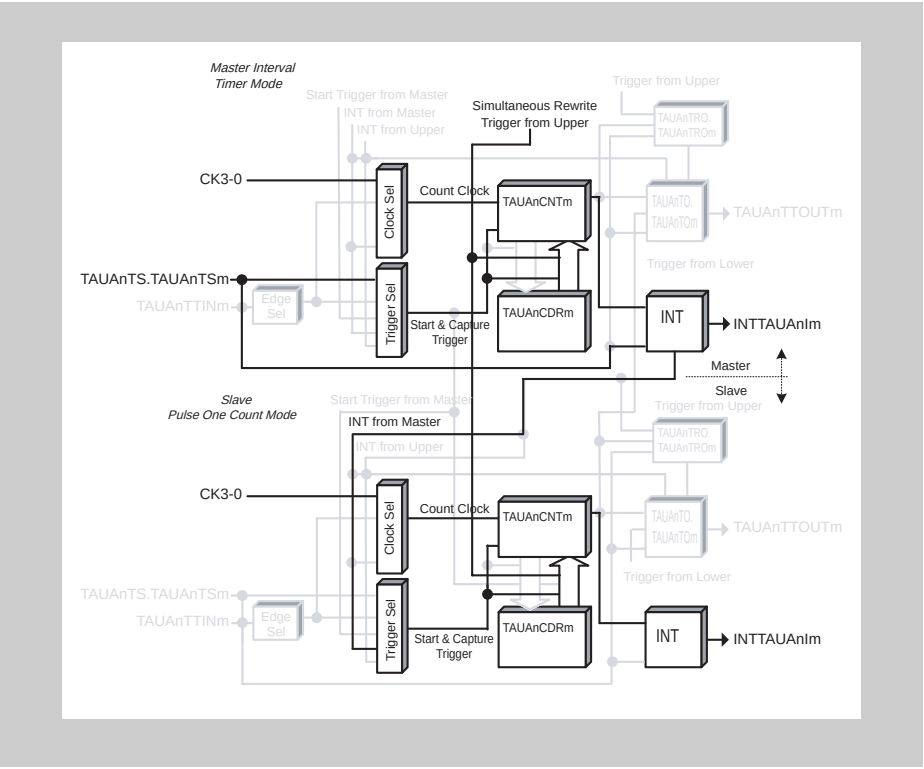


Figure 15-100 Block diagram for AD conversion trigger output function type 1

(3) General timing diagram

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUAnTOL.TAUAnTOLm = 0)

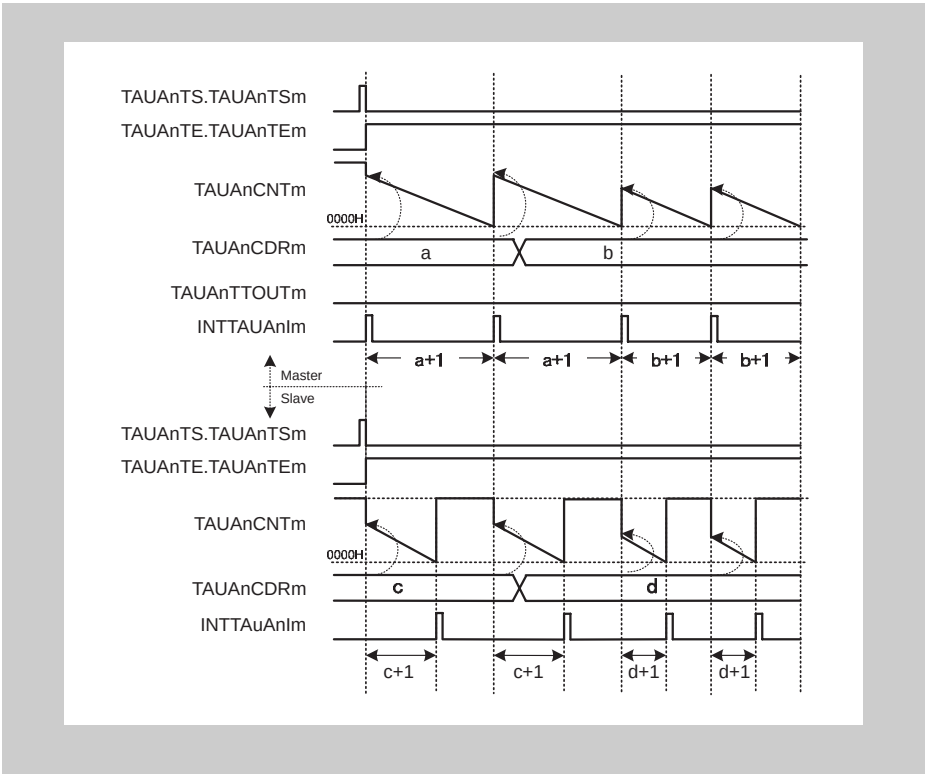


Figure 15-101 General timing diagram for AD conversion trigger output function type 1

15.24 Synchronous PWM Signal Functions Triggered by an External Signal

This chapter describes functions that generate PWM signals and which are triggered by an external signal.

- *15.24.1 “One-shot pulse output function”*
- *15.24.2 “Offset trigger output function”*

15.24.1 One-shot pulse output function

(1) Overview

Summary This function outputs a signal pulse with a defined pulse width and a specific delay time compared to an external input signal pulse by using a master and a slave channel. The delay time is specified using the master channel. The pulse width is specified using the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to one-count mode. See *Table 15-137 “TAUAnCMORm settings for the master channel of the one-shot pulse output function” on page 812*.
- The operation mode of the slave channel must be set to pulse one-count mode. See *Table 15-140 “TAUAnCMORm settings for the slave channel of the one-shot pulse output function” on page 814*.
- TAUAnTTOUTm is not used for the master channel of this function.
- The channel output mode of the slave channel must be set to synchronous channel output mode 2 (see *15.9 “Channel Output Modes” on page 627*).
- TAUAnTTINm (master) has to be detected while TAUAnCNTm (master) and TAUAnCNTm (slave) await a trigger. Furthermore, the slave is only triggered by an interrupt from the master channel and not by TAUAnTTINm (slave).

Description The counters are enabled by setting the channel trigger bits (TAUAnTS.TAUAnTSm) of the master and slave channels to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation.

- Master channel:

  When the next valid TAUAnTTINm input edge is detected, the current value of TAUAnCDRm is loaded to TAUAnCNTm. The counter starts to count down from this value. If TAUAnCMORm.TAUAnMD0 = 0, a trigger (TAUAnTTINm) which is detected within the delay time is ignored.
  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated. The counter returns to $FFFF_H$ and awaits the next valid TAUAnTTINm input edge.

- Slave channel

  The INTTAUAnIm of the master channel triggers the counter of the slave channel. The current value of TAUAnCDRm (slave) is loaded to TAUAnCNTm (slave) and the counter starts to count down from this value. An interrupt is generated and the TAUAnTTOUTm signal is set.
  When the counter reaches 0001_H, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset. The counter stops at 0000_H and awaits the next INTTAUAnIm of the master channel.

The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channel, which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel stop but retain their values. The counters can be restarted by setting TAUAnTS.TAUAnTSm to 1.

The counter of the master channel can be restarted without stopping it first (forced restart) by setting TAUAnTS.TAUAnTSm to 1 during operation.

Conditions

- If TAUAnCMORm.TAUAnMD0 of the master channel is set to 0, during counting detected TAUAnTTINm input edges are ignored.
- Simultaneous rewrite can be used with this function. See *15.8 "Simultaneous Rewrite" on page 615*.

Equations

Delay to input pulse = (TAUAnCDRm (master) + 1) × count clock cycle

Pulse width = (TAUAnCDRm (slave)) × count clock cycle

(2) Block diagram and general timing diagram

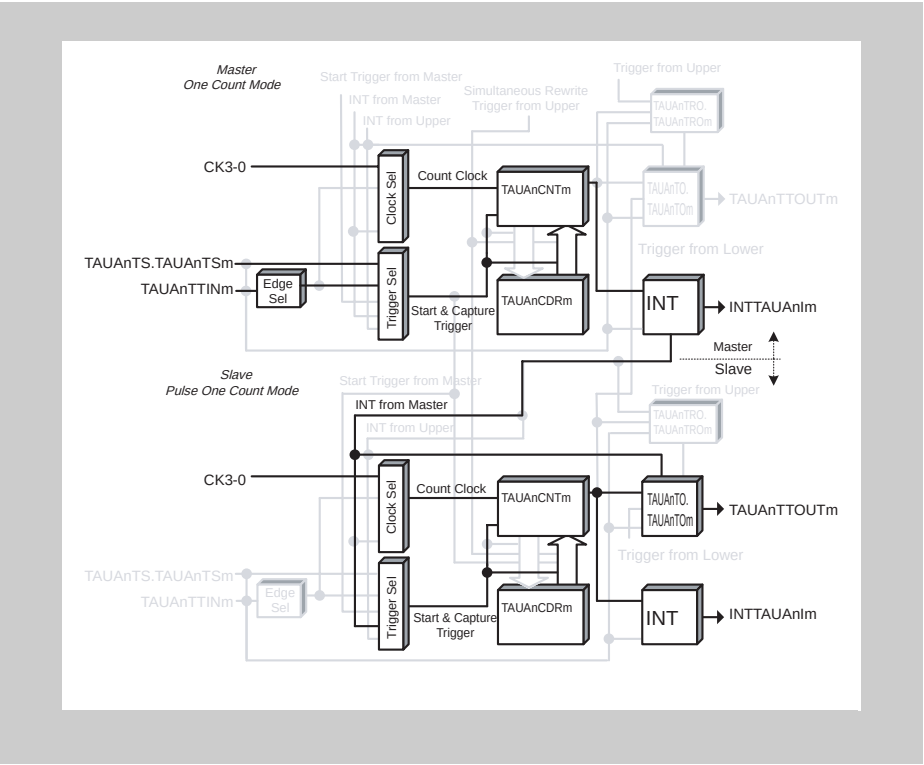


Figure 15-102 Block diagram for one-shot pulse output function

The following settings apply to the general timing diagram:

- Start trigger detection disabled during counting (TAUAnCMORm.TAUAnMD0 = 0)
- Falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)

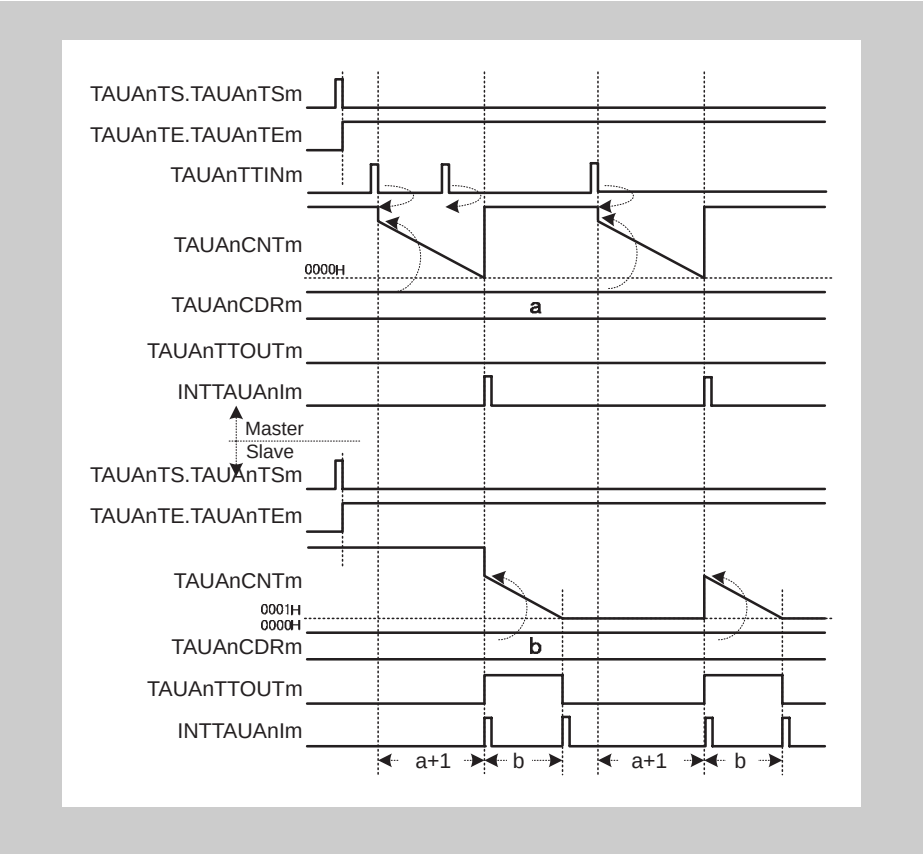


Figure 15-103 General timing diagram for one-shot pulse output function

(3) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-137 TAUAnCMORm settings for the master channel of the one-shot pulse output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	0: Disables start trigger detection during counting 1: Enables start trigger detection during counting The value of the TAUAnMD0 bit of the master and slave channel must be identical.

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-138 TAUAnCMURm settings for the master channel of the one-shot pulse output function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the master channel

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-139 Simultaneous rewrite settings for the master channel of the one-shot pulse output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(4) Register settings for the slave channel

(a) TAUAnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-140 TAUAnCMORm settings for the slave channel of the one-shot pulse output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	1010: Pulse one-count mode
TAUAnMD0	0: Disables start trigger detection during counting 1: Enables start trigger detection during counting The value of the TAUAnMD0 bit of the master and slave channel must be identical.

(b) TAUAnCMURm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-141 TAUAnCMURm settings for the slave channel of the one-shot pulse output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the slave channel

Table 15-142 Control bit settings for synchronous channel output mode 2

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-143 Simultaneous rewrite settings for the slave channel of the one-shot pulse output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is the control channel for simultaneous rewrite
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Operating procedure for one-shot pulse output function

Table 15-144 Operating procedure for one-shot pulse output function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(3) "Register settings for the master channel" on page 812*. Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the slave channel" on page 814*. Set the values of the TAUAnCDRm registers of all channels.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm (master and slave channels) is set to 1 and the master channel awaits a TAUAnTTINm input.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	When the valid edge of TAUAnTTINm input is detected, TAUAnCNTm loads the TAUAnCDRm value of the master channel, and then counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated. • TAUAnCNTm (master) reloads the TAUAnCDRm value, and then continues count operation. • TAUAnCNTm (slave) reloads the TAUAnCDRm value, and then starts counting down. • INTTAUAnIm (slave) is generated. • TAUAnTTOUTm (slave) is set. When TAUAnCNTm (slave) reaches 0001_H: • INTTAUAnIm (slave) is generated. • TAUAnTTOUTm (slave) is reset. If a TAUAnTTINm input is detected on the master channel while the counter is counting, the input is ignored when TAUAnCMORm.TAUAnMD0 = 0.
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart →

15.24.2 Offset trigger output function

(1) Overview

Summary This function generates a PWM output using a master and a slave channel. It enables the pulse width (duration) of the TAUAnTTOUTm to be set. The pulse cycle is set by the detection of a valid input edge of the master channel. The pulse width is specified using the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to capture mode. See *Table 15-145 “TAUAnCMORm settings for the master channel of the offset trigger output function” on page 820*.
- The operation mode of the slave channel must be set to one-count mode. See *Table 15-148 “TAUAnCMORm settings for the slave channel of the offset trigger output function” on page 822*.
- The channel output mode of the slave channel must be set to independent channel output mode 2 (see *15.9 “Channel Output Modes” on page 627*).
- TAUAnTTOUTm is not used for the master channel of this function.

Description The counters are started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation. The counter of the master channel (TAUAnCNTm) starts to count up from 0000_H.

- Master channel:

  When a valid TAUAnTTINm input edge is detected, the current value of the counter (TAUAnCNTm) is loaded to the data register of the master channel (TAUAnCDRm). INTTAUAnIm is generated and the counter restarts counting up from 0000_H.

- Slave channel:

  The INTTAUAnIm of the master channel sets the TAUAnTTOUTm (slave) signal and triggers the counter of the slave channel. The current value of TAUAnCDRm (slave) is loaded to TAUAnCNTm (slave) and the counter starts to count down from this value.

  When the counter reaches 0000_H, i.e. duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel.

The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channel(s), which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel(s) stop but retain their values. The counters can be restarted by setting TAUAnTS.TAUAnTSm to 1.

(2) Equations

<R> Pulse width = TAUAnCDRm (slave) x count clock cycle

Duty cycle [%] = [TAUAnCDRm (slave) / TAUAnTTINm cycle + 1)] x 100

– Duty cycle = 0 %

TAUAnCDRm (slave) = 0000_H

– Duty cycle = 100 %

TAUAnCDRm (slave) ≥ TAUAnTTINm cycle + 1

(3) Block diagram and general timing diagram

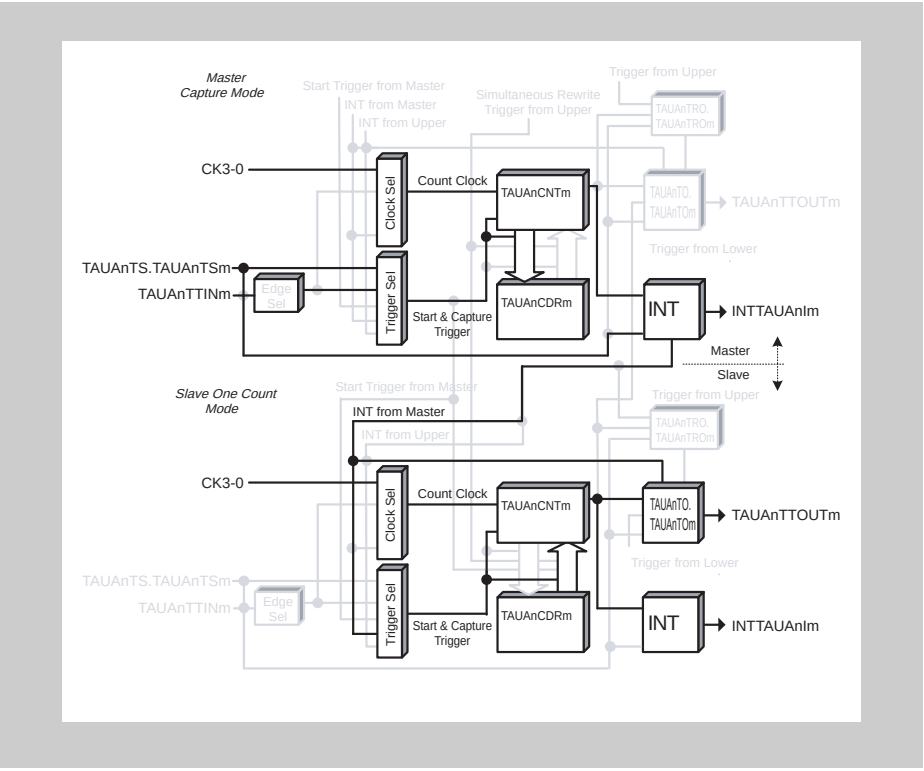


Figure 15-104 Block diagram for offset trigger output function

The following settings apply to the general timing diagram:

- Falling edge detection (TAUAnCMURm.TAUAnTIS[1:0] = 00_B)

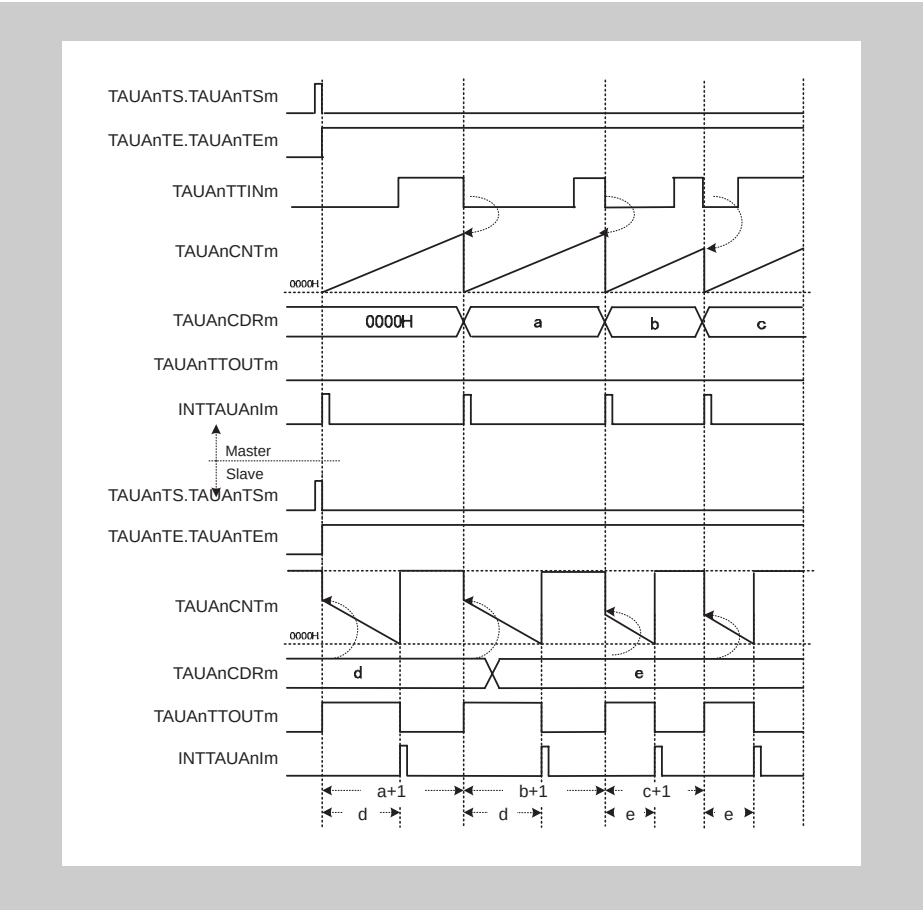


Figure 15-105 General timing diagram for offset trigger output function

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-145 TAUAnCMORm settings for the master channel of the offset trigger output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the start trigger
TAUAnCOS[1:0]	11: Capture register updated upon detection of a TAUAnTTINm input valid edge and upon counter overflow: - • TAUAnTTINm input valid edge: Counter value is written to TAUAnCDRm - • Overflow: $FFFF_H$ is written to TAUAnCDRm. The next TAUAnTTINm input valid edge detection is ignored. TAUAnCSRm.TAUAnOVF set by counter overflow and cleared by a CPU instruction (by setting the TAUAnCSCm.TAUAnCLOV bit to 1).
TAUAnMD[4:1]	0010: Capture mode
TAUAnMD0	1: Generates INTTAUAnIm at operation start

<R>

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-146 TAUAnCMURm settings for the master channel of the offset trigger output function

Bit name	Setting
TAUAnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the master channel

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the offset trigger output function. Therefore, these registers must be set to 0.

Table 15-147 Simultaneous rewrite settings for the master channel of the offset trigger output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	When simultaneous rewrite is disabled (TAUAnTAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(5) Register settings for the slave channel

(a) TAUAnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-148 TAUAnCMORm settings for the slave channel of the offset trigger output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Enables start trigger detection during counting

(b) TAUAnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-149 TAUAnCMURm settings for the slave channel of the offset trigger output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the slave channel

Table 15-150 Control bit settings for independent channel output mode 2

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output operation
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the offset trigger output function. Therefore, these registers must be set to 0.

Table 15-151 Simultaneous rewrite settings for the slave channel of the offset trigger output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	0: Disables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	When simultaneous rewrite is disabled (TAUAnTAUAnRDE.TAUAnRDEm = 0), set these bits to 0
TAUAnRDM.TAUAnRDMm	
TAUAnRDC.TAUAnRDCm	

(6) Operating procedure for offset trigger output function

Table 15-152 Operating procedure for offset trigger output function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 820*. Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for the slave channel" on page 822*. Set the values of the TAUAnCDRm registers of all channels.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm (master and slave channels) is set to 1 and the counters of the master and slave channels start: • TAUAnCNTm (master) counts up. • TAUAnCNTm (slave) loads TAUAnCDRm, and then counts down. INTTAUAnIm is generated on the master channel and TAUAnTTOUTm (master) is set.
During operation	TAUAnCDRm can be changed at any time. TAUAnCSCm.TAUAnCLOV can be set to 1.	When TAUAnCNTm of the slave reaches 0000_H: • INTTAUAnIm (slave) is generated. • TAUAnTTOUTm (slave) is reset. When a TAUAnTTINm input is detected on the master channel: • INTTAUAnIm (master) is generated. • TAUAnCNTm (master) is reset to 0000_H and continues count operation. • TAUAnCNTm (slave) reloads the TAUAnCDRm value, and then counts down. • TAUAnTTOUTm (slave) is set.
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

15.25 Synchronous Triangle PWM Functions

This chapter describes functions that generate a triangle PWM output.

- *15.25.1 “Triangle PWM output function”*
- *15.25.2 “Triangle PWM output function with dead time”*
- *15.25.3 “AD conversion trigger output function type 2”*

15.25.1 Triangle PWM output function

(1) Overview

Summary This function generates multiple triangle PWM outputs by using a master and one or more slave channels. It enables the pulse cycle (frequency) and the duty cycle of TAUAnTTOUTm to be set using the master and slave channel(s) respectively.

Carrier cycles are generated on the master channel. The first pulse of the master channel controls the down status and the second pulse controls the up status of the slaves counter.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to interval timer mode. See *Table 15-153 "TAUAnCMORm settings for the master channel of the triangle PWM output function" on page 830*.
- The operation mode of the slave channel(s) must be set to up/down count mode. See *Table 15-157 "TAUAnCMORm settings for the slave channel of the triangle PWM output function" on page 832*.
- The channel output mode of the master channel must be set to independent channel output mode 1 (see *15.9 "Channel Output Modes" on page 627*).
- The channel output mode of the slave channel(s) must be set to synchronous channel output mode 2 (see *15.9 "Channel Output Modes" on page 627*).
- The following settings establish TAUAnTTOUTm at high level for the down status of the carrier cycle.
  - If the TAUAnCMORm.TAUAnMD0 (master) bit is set to 0, TAUAnTO.TAUAnTOm must be set to 1 while TAUAnTOE.TAUAnTOEm is 0 (recommended).
  - If the TAUAnCMORm.TAUAnMD0 (master) bit is set to 1, TAUAnTO.TAUAnTOm must be set to 0 while TAUAnTOE.TAUAnTOEm is 0.

Description The counters are started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1 for every channel. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation. The values of TAUAnCDRm (master and slave) are loaded to TAUAnCNTm (master and slave) and the counters start to count down from these values. If the master channel TAUAnCMORm.TAUAnMD0 bit is set, an interrupt is generated and TAUAnTTOUTm signal of the master toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal toggles. TAUAnCNTm then reloads the TAUAnCDRm value, and then counts down.

- Slave channel:

  The INTTAUAnIm of the master channel triggers the counter of the slave channel:

  – If the slave counter currently counts down, it changes count direction.

  – If the slave counter currently counts up, the value of TAUAnCDRm is reloaded and the counter counts down.

  When the counter of the slave channel reaches 0001_H while counting up or down, INTTAUAnIm is generated and the TAUAnTTOUTm (slave) signal is set or reset.

  The counter continues to count down or up and awaits the next INTTAUAnIm of the master channel.

TAUAnTTOUTm can be switched between positive and negative phase setting TAUAnTOL.TAUAnTOLm during operation.

The counters can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channel(s), which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel(s) stop but retain their values.

Conditions Simultaneous rewrite can be used with this function. See *15.8 “Simultaneous Rewrite” on page 615*.

(2) Equations

Pulse cycle = (TAUAnCDRm (master) + 1) x count clock cycle

$0000_H \leq$ TAUAnCDR (master) $< FFFF_H$

Carrier cycle (down/up) = (TAUAnCDRm (master) + 1) × 2 × count clock cycle

Duty cycle [%] =
[(TAUAnCDRm (master) + 1 - TAUAnCDRm (slave))/ (TAUAnCDRm (master) + 1)] x 100

– Duty cycle = 100 %

  TAUAnCDRm (slave) = 0000_H

– Duty cycle = 0 %

  TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

(3) Block diagram and general timing diagram

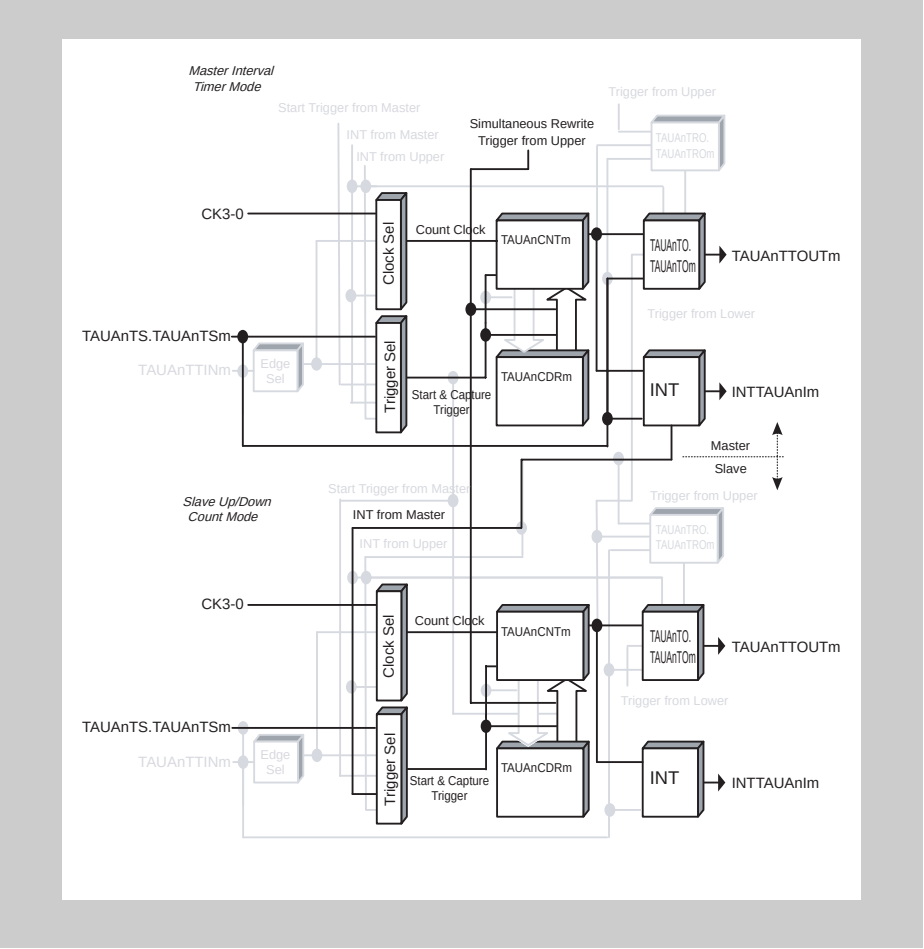


Figure 15-106 Block diagram for triangle PWM output function

The following settings apply to the general timing diagram:

- Master channel
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)

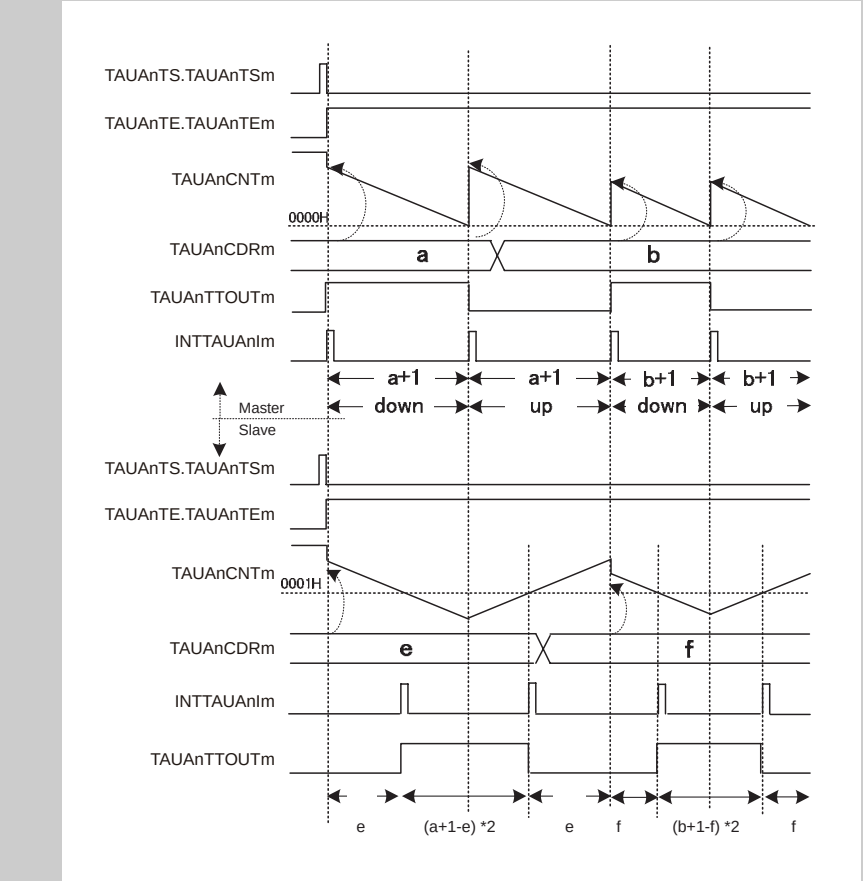


Figure 15-107 General timing diagram for triangle PWM output function

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-153 TAUAnCMORm settings for the master channel of the triangle PWM output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-154 TAUAnCMURm settings for the master channel of the triangle PWM output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the master channel

Table 15-155 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-156 Simultaneous rewrite settings for the master channel of the triangle PWM output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

Note If the TAUAnRDS.TAUAnRDSm bit is 1, there must be a channel higher than the master channel to which a simultaneous rewrite trigger signal is generated.

(5) Register settings for the slave channel(s)

(a) TAUAnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-157 TAUAnCMORm settings for the slave channel of the triangle PWM output function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	111: The up/down output trigger signal of the master channel
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	1001: Up/down count mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-158 TAUAnCMURm settings for the slave channel of the triangle PWM output function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the slave channel(s)

Table 15-159 Control bit settings for synchronous channel output mode 2

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel operation
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-160 Simultaneous rewrite settings for the slave channel of the triangle PWM output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for triangle PWM output function

Table 15-161 Operating procedure for triangle PWM output function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 830*. Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for the slave channel(s)" on page 832*. Set the values of the TAUAnCDRm registers of all channels.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. If TAUAnCMORm.TAUAnMD0 is set to 1 on the master channel, INTTAUAnIm (master) is generated.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	TAUAnCNTm loads the TAUAnCDRm value of the master channel or slave channel, and then counts down. When the counter of the master channel reaches 0000_H: • INTTAUAnIm (master) is generated. • TAUAnTTOUTm (master) toggles. • TAUAnCNTm (master) reloads the TAUAnCDRm value, and then continues count operation. • TAUAnCNTm (slave) reloads the TAUAnCDRm value or counts in the reverse direction. When TAUAnCNTm of the slave = 0001_H: • INTTAUAnIm (slave) is generated. • TAUAnTTOUTm (slave) is set (in count-down status) or reset (in count-up status).
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0%

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)
  - TAUAnCDRm = a = 5_H
- Slave channel:
  - TAUAnCDRm = 6_H

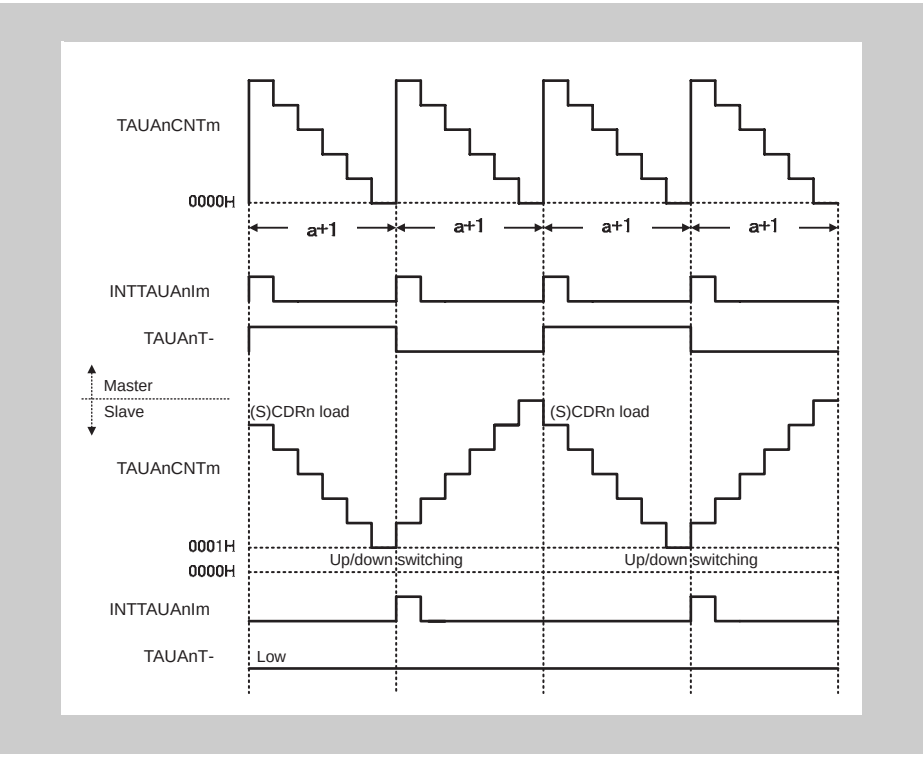


Figure 15-108 TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

- If TAUAnCDRm (slave) ≥ TAUAnCDRm (master) the counter of slave channel cannot reach 0001_H during *counting down*. The set signal is never detected, so TAUAnTTOUTm remains at low state.

(b) Duty cycle = 100%

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)
  - TAUAnCDRm = a = 5_H
- Slave channel:
  - TAUAnCDRm = 0_H

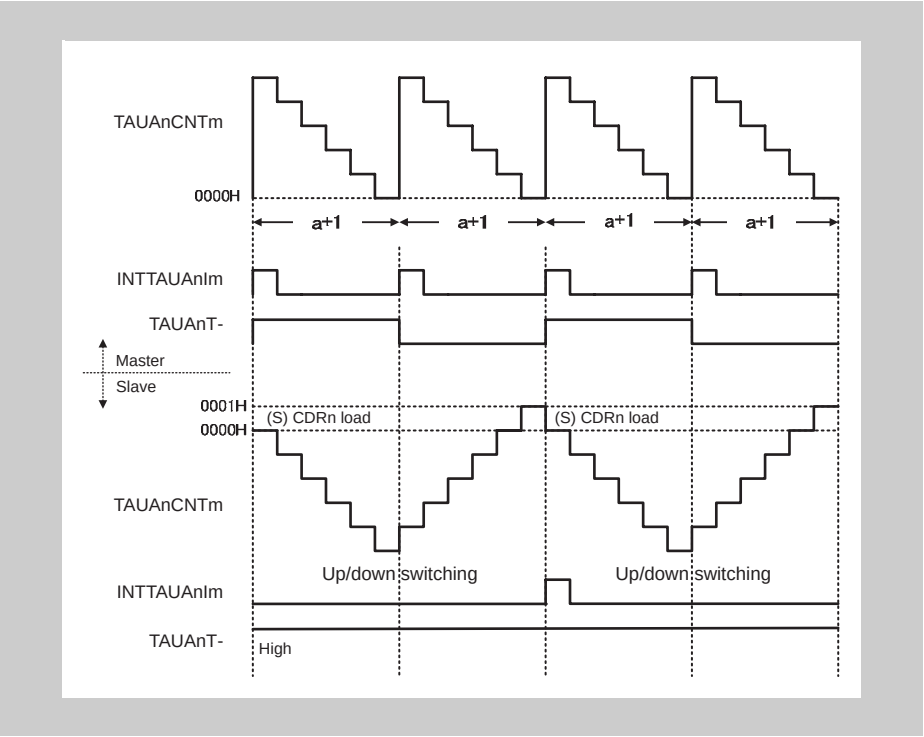


Figure 15-109 TAUAnCDRm (slave) = 0000_H

- If TAUAnCDRm (slave) =0000_H the counter of slave channel cannot reach 0001_H during *counting up*. The reset signal is never detected, so TAUAnTTOUTm remains at high state.

15.25.2 Triangle PWM output function with dead time

(1) Overview

Summary This function generates multiple triangle PWM outputs with a defined dead time by using a master and two or more slave channels. The resulting PWM signals are output via TAUAnTTOUTm of the slave channel 3. It enables the pulse cycle (frequency) and the duty cycle of TAUAnTTOUTm to be set using the master and slave channel(s) respectively.

Carrier cycles are generated on the master channel. The first pulse cycle controls the down status of the slave counter, and the second pulse cycle controls the up status.

An interrupt on slave 2 causes TAUAnTTOUTm of the slave channels to be set or reset. Depending on the settings of TAUAnTDL.TAUAnTDLm, delay time is added to positive or negative logic side of the signal (i.e. whether TAUAnTTOUTm is set/reset immediately or after dead time has elapsed). The duration of the dead time is specified by slave channel 3.

Prerequisites

- Three channels. For slave channels 2 and 3, select the even channel CH (a) and the odd channel CH (a + 1).
- The operation mode of the master channel must be set to interval timer mode. See *Table 15-163 “TAUAnCMORm settings for the master channel of the triangle PWM output function with dead time” on page 842*.

<R>

- Slave channel 1 is not used for this function. This ensures that slave channel 2 is an even channel, and slave channel 3 is an odd channel.
- The operation mode of slave channel 2 must be set to Up Down Mode. See *Table 15-167 “TAUAnCMORm settings for slave channel 2 of the triangle PWM output function with dead time” on page 844*.
  Furthermore, slave channel 2 must be an even channel.
- The operation mode of slave channel 3 must be set to one-count mode. See *Table 15-171 “TAUAnCMORm settings for slave channel 3 of the triangle PWM output function with dead time” on page 846*.
  Furthermore, slave channel 3 must be an odd channel.
- The channel output mode of the master channel must be set to independent channel output mode 1 (see *15.9 “Channel Output Modes” on page 627*).
- The channel output mode of slave channels 2 and 3 must be set to synchronous channel output mode 2 with dead time output (see *15.9 “Channel Output Modes” on page 627*).
- The following settings establish TAUAnTTOUTm at high level for the down status of the carrier cycle.
  - If the TAUAnCMORm.TAUAnMD0 (master) bit is set to 0, TAUAnTO.TAUAnTOm must be set to 1 while TAUAnTOE.TAUAnTOEm is 0 (recommended).
  - If the TAUAnCMORm.TAUAnMD0 (master) bit is set to 1, TAUAnTO.TAUAnTOm must be set to 0 while TAUAnTOE.TAUAnTOEm is 0.

Note Slave channel 1 is not used for triangle PWM output function with dead time.

Description

The counters are started by setting the channel trigger bits (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation. The current values of TAUAnCDRm is loaded to TAUAnCNTm and the counters start to count down from these values. If the master channel TAUAnCMORm.TAUAnMD0 bit is set, an interrupt is generated and TAUAnTTOUTm signal of the master toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated and the TAUAnTTOUTm signal toggles. The counter reloads the TAUAnCDRm value, and then counts down.

- Slave channel 2:

  The INTTAUAnIm of the master channel triggers the counter of the slave channel 2:

  – If the slave counter currently counts down, it changes count direction.

  – If the slave counter currently counts up, the value of TAUAnCDRm is reloaded and the counter counts down.

  The counter continues to count down or up and awaits the next INTTAUAnIm of the master channel.

- Slave channel 3:

  INTTAUAnIm of slave channel 2 triggers the counter of slave channel 3. The current value of TAUAnCDRm (slave 3) is loaded to TAUAnCNTm (slave 3) and the counter starts to count down from this value.

  When the counter reaches 0000_H, INTTAUAnIm is generated. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of slave channel 2.

The TAUAnTDL.TAUAnTDLm settings of the corresponding channel specify whether it is set/reset immediately, or after dead time has elapsed, as shown in *Table 15-162 “Behavior of TAUAnTTOUTm when an interrupt occurs on slave channel 2” on page 839*.

The TAUAnTOL.TAUAnTOLm settings specify whether set corresponds to a high signal (TAUAnTOL.TAUAnTOLm = 0) or a low signal (TAUAnTOL.TAUAnTOLm = 1).

The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channel(s), which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel(s) stop but retain their values.

TAUAnCDRm value of slave channel 2 can be set to 0000_H to output 100 % TAUAnTTOUTm.

Conditions Simultaneous rewrite can be used with this function. See *15.8 "Simultaneous Rewrite" on page 615*.

TAUAnTOL.TAUAnTOLm and TAUAnTDL.TAUAnTDLm bits should be set before the counter starts, and slave channels 2 and 3 should have opposite TAUAnTOL.TAUAnTOLm settings or opposite TAUAnTDL.TAUAnTDLm settings.

Table 15-162 Behavior of TAUAnTTOUTm when an interrupt occurs on slave channel 2

TAUAnTDL. TAUAnTDLm	Count direction of slave channel 2 when interrupt is generated	TAUAnTTOUTm set/reset timing
0	Down	Set after dead time has elapsed
	Up	Reset immediately
1	Down	Set immediately
	Up	Reset after dead time has elapsed

(2) Equations

Pulse cycle = (TAUAnCDRm (master) + 1) x count clock cycle

$0000_H \leq$ TAUAnCDR (master) $< FFFF_H$

Carrier cycle (down/up) = (TAUAnCDRm (master) + 1) × 2 × count clock cycle

PWM signal width (positive phase) = [((TAUAnCDRm (master) + 1 - TAUAnCDRm (slave 2) × 2) - (TAUAnCDRm (slave 3) + 1)] × count clock cycle

PWM signal width (negative phase) = [((TAUAnCDRm (master) + 1 - TAUAnCDRm (slave 2) × 2) + (TAUAnCDRm (slave 3) + 1)] × count clock cycle

(3) Block diagram and general timing diagram

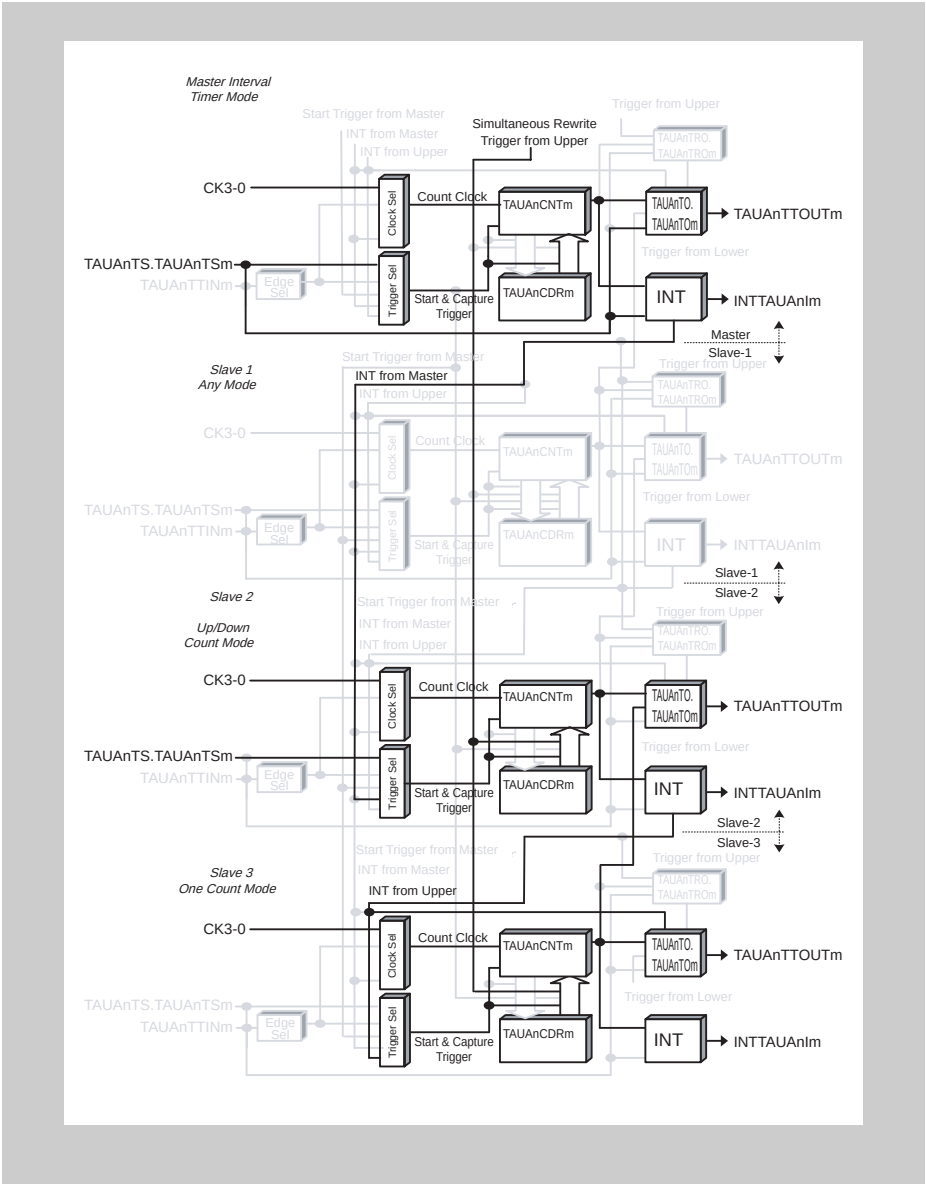


Figure 15-110 Block diagram for triangle PWM output function with dead time

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)
- Slave channel 2:
  - INTTAUAnIm not generated at operation start (TAUAnCMORm.TAUAnMD0 = 0)
  - TAUAnTDL.TAUAnTDLm = 0
  - Positive logic (TAUAnTOL.TAUAnTOLm = 0)
- Slave channel 3:
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)
  - TAUAnTDL.TAUAnTDLm = 1
  - Negative logic (TAUAnTOL.TAUAnTOLm = 1)

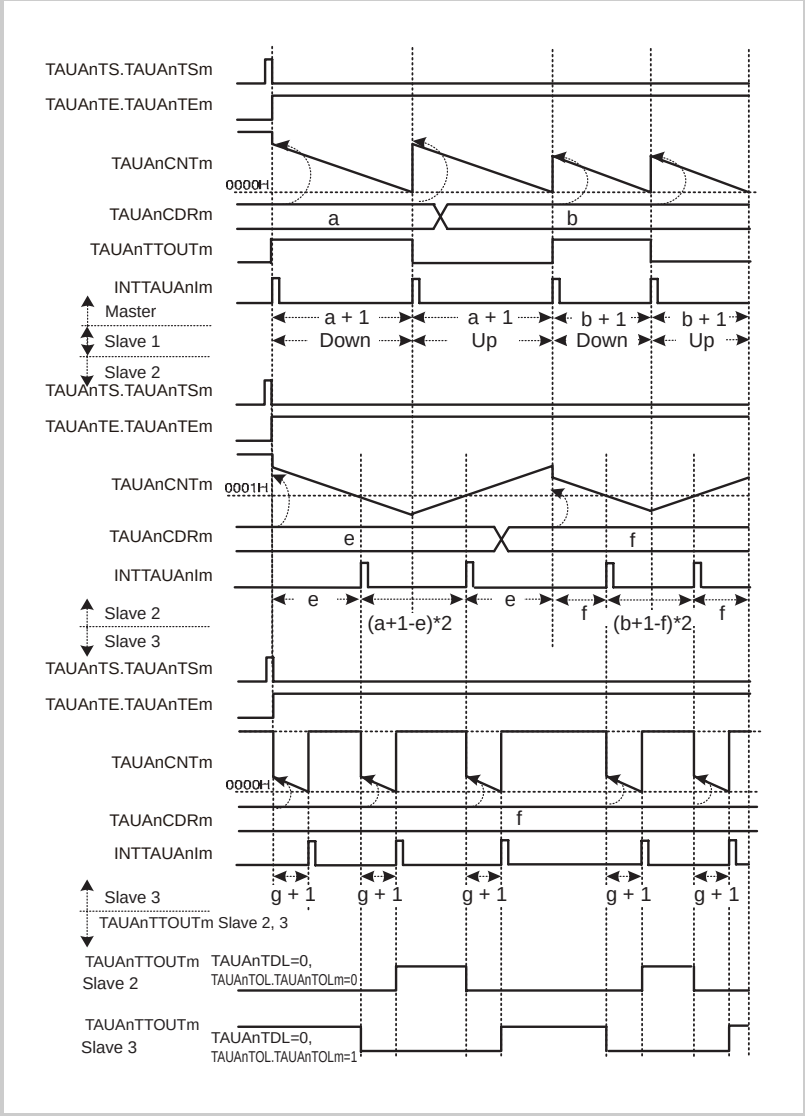


Figure 15-111 General timing diagram for triangle PWM output function with dead time

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAn STS[2:0]			TAUAn COS[1:0]		-	TAUAn MD[4:1]				TAUAn MD0

Table 15-163 TAUAnCMORm settings for the master channel of the triangle PWM output function with dead time

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-164 TAUAnCMURm settings for the master channel of the triangle PWM output function with dead time

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the master channel

Table 15-165 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-166 Simultaneous rewrite settings for the master channel of the triangle PWM output function with dead time

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

Note If the TAUAnRDS.TAUAnRDSm bit is 1, there must be a channel higher than the master channel to which a simultaneous rewrite trigger signal is generated.

(5) Register settings for slave channel 2

(a) TAUAnCMORm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-167 TAUAnCMORm settings for slave channel 2 of the triangle PWM output function with dead time

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	111: The up/down output trigger signal of the master channel
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	1001: Up/down count mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-168 TAUAnCMURm settings for slave channel 2 of the triangle PWM output function with dead time

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for slave channel 2

Table 15-169 Control bit settings for synchronous channel output mode 2 with dead time output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel operation
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	1: Enables dead time operation
TAUAnTDM.TAUAnTDMm	0: Dead time is added upon detection of an interrupt on the upper even channel, if the TAUAnTDL.TAUAnTDLm condition is also fulfilled.
TAUAnTDL.TAUAnTDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

Caution For TAUAnTDL.TAUAnTDLm, specify the setting that is opposite that of the odd channel.

(d) Simultaneous rewrite for slave channel 2

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-170 Simultaneous rewrite settings for slave channel 2 of the triangle PWM output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channel 3

(a) TAUAnCMORm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-171 TAUAnCMORm settings for slave channel 3 of the triangle PWM output function with dead time

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	110: Dead time trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Enables start trigger detection during counting

(b) TAUAnCMURm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-172 TAUAnCMURm settings for slave channel 3 of the triangle PWM output function with dead time

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for slave channel 3

Table 15-173 Control bit settings for synchronous channel output mode 2 with dead time output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel operation
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	1: Enables dead time operation
TAUAnTDM.TAUAnTDMm	0: Dead time is added upon detection of an interrupt on the upper even channel, if the TAUAnTDL.TAUAnTDLm condition is also fulfilled.
TAUAnTDL.TAUAnTDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

Caution For TAUAnTDL.TAUAnTDLm, specify the setting that is opposite that of the even channel.

(d) Simultaneous rewrite for slave channel 3

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-174 Simultaneous rewrite settings for slave channel 3 of the triangle PWM output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for triangle PWM output function with dead time

Table 15-175 Operating procedure for triangle PWM output function with dead time

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 842*. Slave channel 2: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for slave channel 2" on page 844*. Slave channel 3: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(6) "Register settings for slave channel 3" on page 846*. Set the values of the TAUAnCDRm registers of all channels.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUAnIm is generated on the master channel, if TAUAnCMORm.TAUAnMD0 of the master channel is set.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	TAUAnCNTm loads the TAUAnCDRm value of the master channel and slave channel 2, and then counts down. When the counter of the master channel reaches 0000_H: • INTTAUAnIm (master) is generated. • TAUAnCNTm (master) reloads the value of TAUAnCDRm, and then continues counting. • TAUAnCNTm (slave 2) reloads the value of TAUAnCDRm, or starts counting in the opposite direction. When TAUAnCNTm (slave 2) reaches 0001_H: • INTTAUAnIm (slave 2) is generated. • TAUAnCNTm loads the TAUAnCDRm value of slave channel 3, and then counts down. When TAUAnCNTm of slave channel 3 = 0000_H: • INTTAUAnIm is generated.
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart →

(8) Specific timing diagrams

(a) Duty cycle = 0%

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUAnTOL.TAUAnTOLm = 0)
- Slave channel 3:
  - Negative logic (TAUAnTOL.TAUAnTOLm = 1)

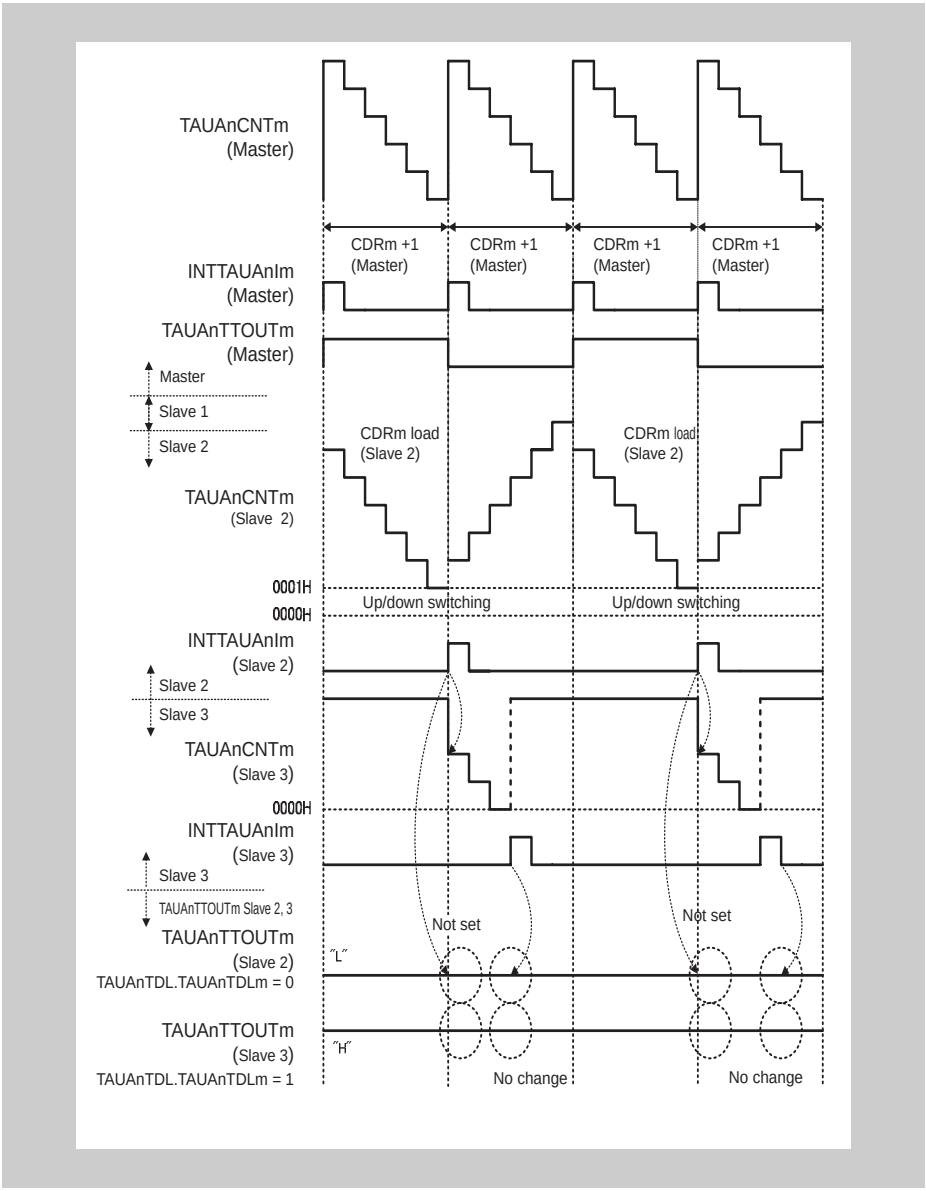


Figure 15-112 TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

- If TAUAnCDRm (slave 2) ≥ TAUAnCDRm (master) the counter of slave channel cannot reach 0000_H during counting down. Therefore TAUAnTTOUTm cannot be set or reset, and it remains at its initial state. The interrupt from slave channel 2 occurs during count up, therefore it is a reset signal.

(b) Duty cycle = 100%

The following settings apply to the diagram below:

- Slave channels 2:
  - Positive logic (TAUAnTOL.TAUAnTOLm = 0)
- Slave channels 3:
  - Negative logic (TAUAnTOL.TAUAnTOLm = 1)

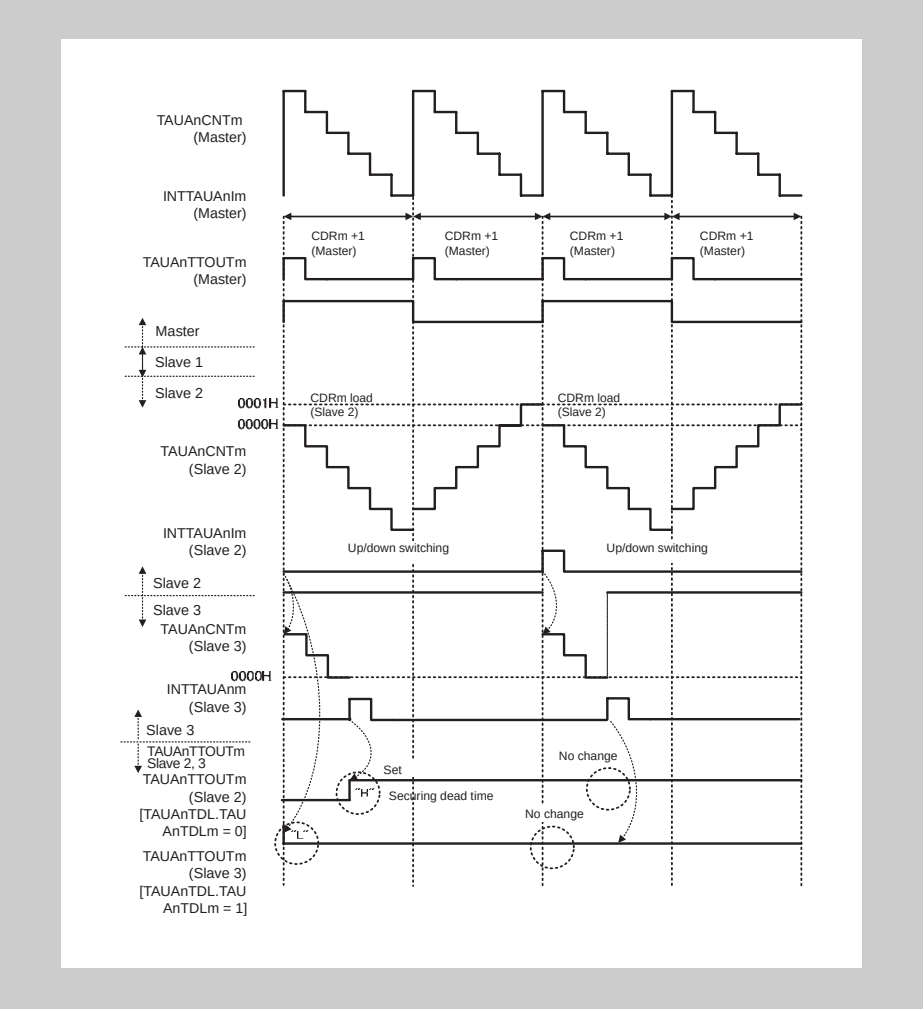


Figure 15-113 TAUAnCDRm (slave) = 0000_H

- If TAUAnCDRm (slave 2) =0000_H the counter of slave channel cannot reach 0001_H while counting up and therefore no INTTAUAnIm is generated while counting up.
  - The set conditions for a channel in which TAUAnTDL.TAUAnTDLm = 0 are met after dead time has elapsed. TAUAnTTOUTm is set/reset but remains in the new state because the reset conditions never occur for such a channel.
  - Slave channel 3 in the diagram above is set when the counter starts. However, the reset conditions for a channel in which TAUAnTDL.TAUAnTDLm = 1 never occur so TAUAnTTOUTm remains in its initial state for such a slave channel.

15.25.3 AD conversion trigger output function type 2

(1) Overview

<R> **Summary** This is achieved by setting the channel output mode of the slave to independent channel output mode controlled by software.

(2) Block diagram and general timing diagram

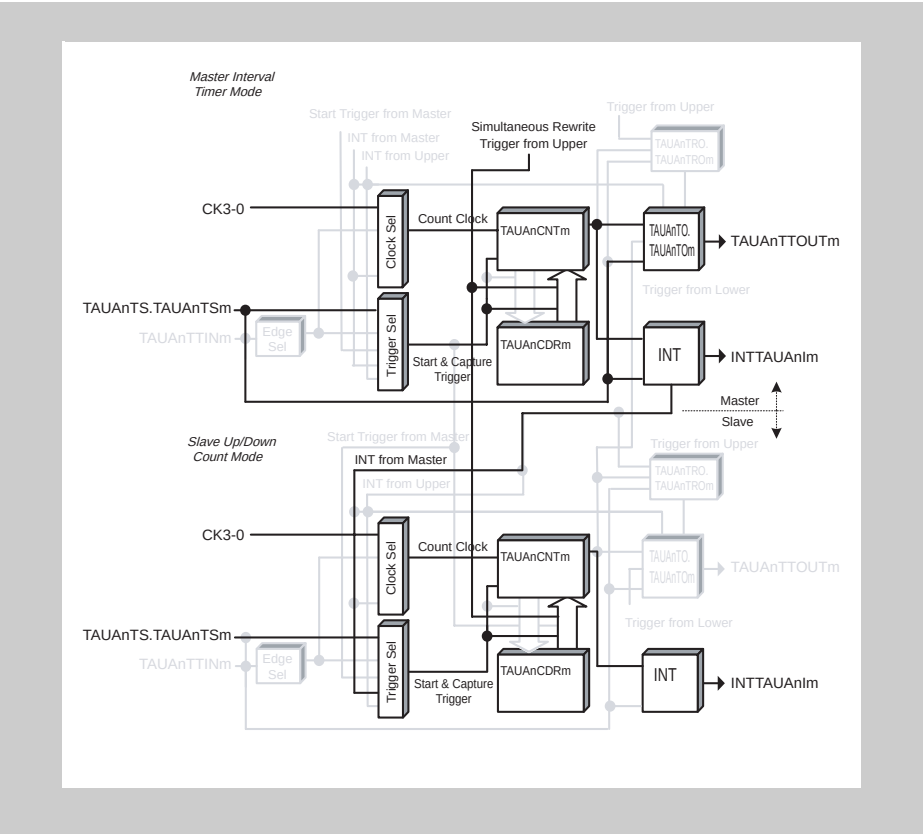


Figure 15-114 Block diagram for AD conversion trigger output function type 2

The following settings apply to the general timing diagram:

- Master channel
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)

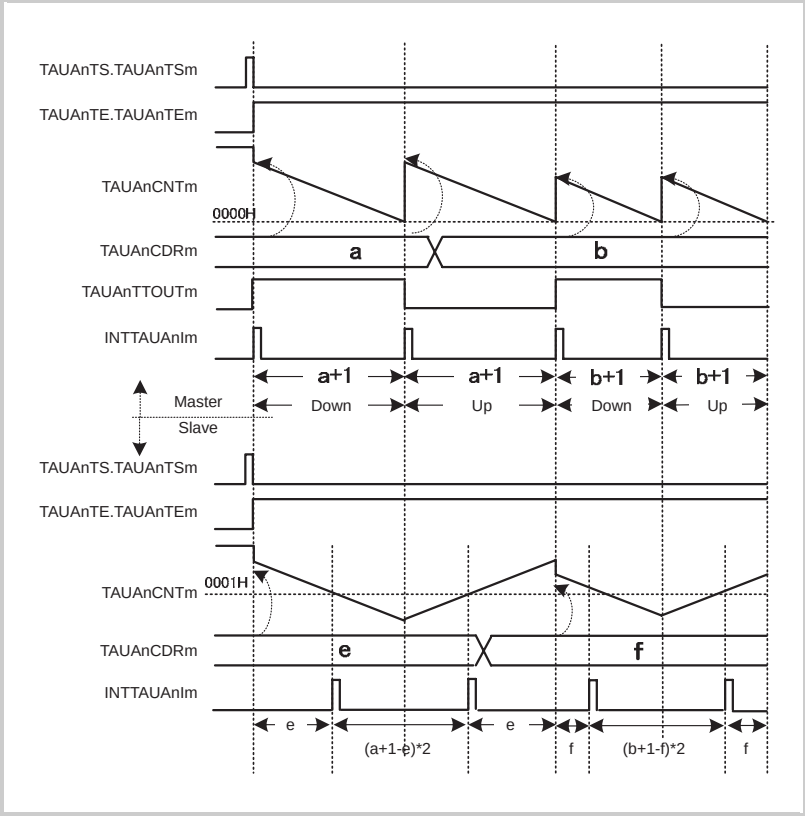


Figure 15-115 General timing diagram for AD conversion trigger output function type 2

15.26 Synchronous Non-Complementary and Complementary Functions

This chapter describes functions that generate 6-phase triangle PWM using a master channel and 7 slaves.

- *15.26.1 “Non-complementary modulation output function type 1”*
- *15.26.2 “Non-complementary modulation output function type 2”*
- *15.26.3 “Complementary modulation output function”*

15.26.1 Non-complementary modulation output function type 1

(1) Overview

Summary This function outputs a PWM signal, a high signal, or a low signal from TAUAnTTOUTm depending on the value of the real-time output bits (TAUAnTRO.TAUAnTROm) and the modulation output enable bits (TAUAnTME.TAUAnTMEm) of a pair of slave channels. Three pairs of channels are typically used.

Prerequisites

- One master channel and seven slave channels
- The operation mode of the master channel must be set to interval timer mode. See *Table 15-177 "TAUAnCMORm settings for the master channel of non-complementary modulation output function type 1" on page 858*.
- The operation mode of slave channels 1 to 7 must be set to one-count mode. See *Table 15-180 "TAUAnCMORm settings for slave channel 1 of non-complementary modulation output function type 1" on page 860*.
- TAUAnTTOUTm is not used for the master channel of this function.
- TAUAnTTOUTm of slave channel 1 is not used for this function, but TAUAnTRC.TAUAnTRCm must be set to 1. See *15.9 "Channel Output Modes" on page 627*.
- The channel output mode of slave channels 2 to 7 must be set to Simultaneous Channel Output Mode 1 with non-complementary modulation output. See *15.9 "Channel Output Modes" on page 627*.
- TAUAnCDRm of slave channel 1 must be set to 0000_H.

Description The counters of the master and slave channels are started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation. The values of the data registers (TAUAnCDRm) are loaded to the counters (TAUAnCNTm) and the counters start to count down. When the counters reach 0000_H, INTTAUAnIm is generated.

- Slave channel 1:

  Because slave channel 1 is specified as the trigger channel for real-time output (TAUAnTRC.TAUAnTRCm = 1), when an interrupt is generated on slave channel 1 (for which TAUAnCDRm is fixed to 0), the real-time output bit (TAUAnTRO.TAUAnTROm) value of the channel that detects the generated interrupt of the corresponding channel is changed. After generating an interrupt, the counter returns to $FFFF_H$ and awaits the next interrupt from the master channel.

- Slave channel 2:

  Slave channel 2 generates a PWM output. The PWM output cycle is specified for the master channel, and the duty cycle is specified for slave channel 2. After generating an interrupt, the counter returns to $FFFF_H$ and awaits the next interrupt from the master channel.

Slave channels 3 to 7 behave analogously to slave channel 2.

The signal that is output from TAUAnTTOUTm depends on the value of the real-time output bit (TAUAnTRO.TAUAnTROm) and the modulation output bit (TAUAnTME.TAUAnTMEm) of the slave channel, as shown in *Table 15-176 "TAUAnTTOUTm output of a pair of slave channels in non-complementary modulation output function type 1" on page 855*.

Forced restart is not possible with this function. The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TAUAnTSm to 1.

Conditions

- If TAUAnTME.TAUAnTMEm = 0 for slave channels 2 to 7:
  - If TAUAnTRO.TAUAnTROm of the channel is 1, TAUAnTTOUTm outputs a high signal
  - If TAUAnTRO.TAUAnTROm of the channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTME.TAUAnTMEm = 1 for slave channels 2 to 7:
  - If TAUAnTRO.TAUAnTROm of the channel is 1, TAUAnTTOUTm outputs the corresponding PWM of the channel
  - If TAUAnTRO.TAUAnTROm of the channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTOL.TAUAnTOLm = 1 the high and low signals output from TAUAnTTOUTm are inverted. The PWM signals remain unaffected. Only the initial setting can be used for TAUAnTOL.TAUAnTOLm (and this setting cannot be changed during operation).

Table 15-176 TAUAnTTOUTm output of a pair of slave channels in non-complementary modulation output function type 1

TAUAnTME.TAUAnTMEm	TAUAnTRO.TAUAnTROm	TAUAnTTOUTm outputs
0	0	Low
	1	High
1	0	Low
	1	PWMm

- Simultaneous rewrite can be used with this function. See *15.8 "Simultaneous Rewrite" on page 615*.
- The value of TAUAnCDRm of slave channel 1 must be set to 0000_H so that the real-time output trigger occurs at the same time as the PWM is generated by slave channels 2 to 7.
- If TAUAnTOL.TAUAnTOLm is cleared to 0 on slave channels 2 to 7, TAUAnTO.TAUAnTOm is set to the low level before clearing TAUAnTE.TAUAnTEm to 0.
- If TAUAnTOL.TAUAnTOLm is set to 1 on slave channels 2 to 7, TAUAnTO.TAUAnTOm is set to the high level before clearing TAUAnTE.TAUAnTEm to 0.

(2) Equations

For slave channels 2 to 7:

PWM output cycle time = [TAUAnCDRm (master) + 1] x count clock

PWM output duty time = [TAUAnCDRm (slave)] x count clock

(3) Block diagram and general timing diagram

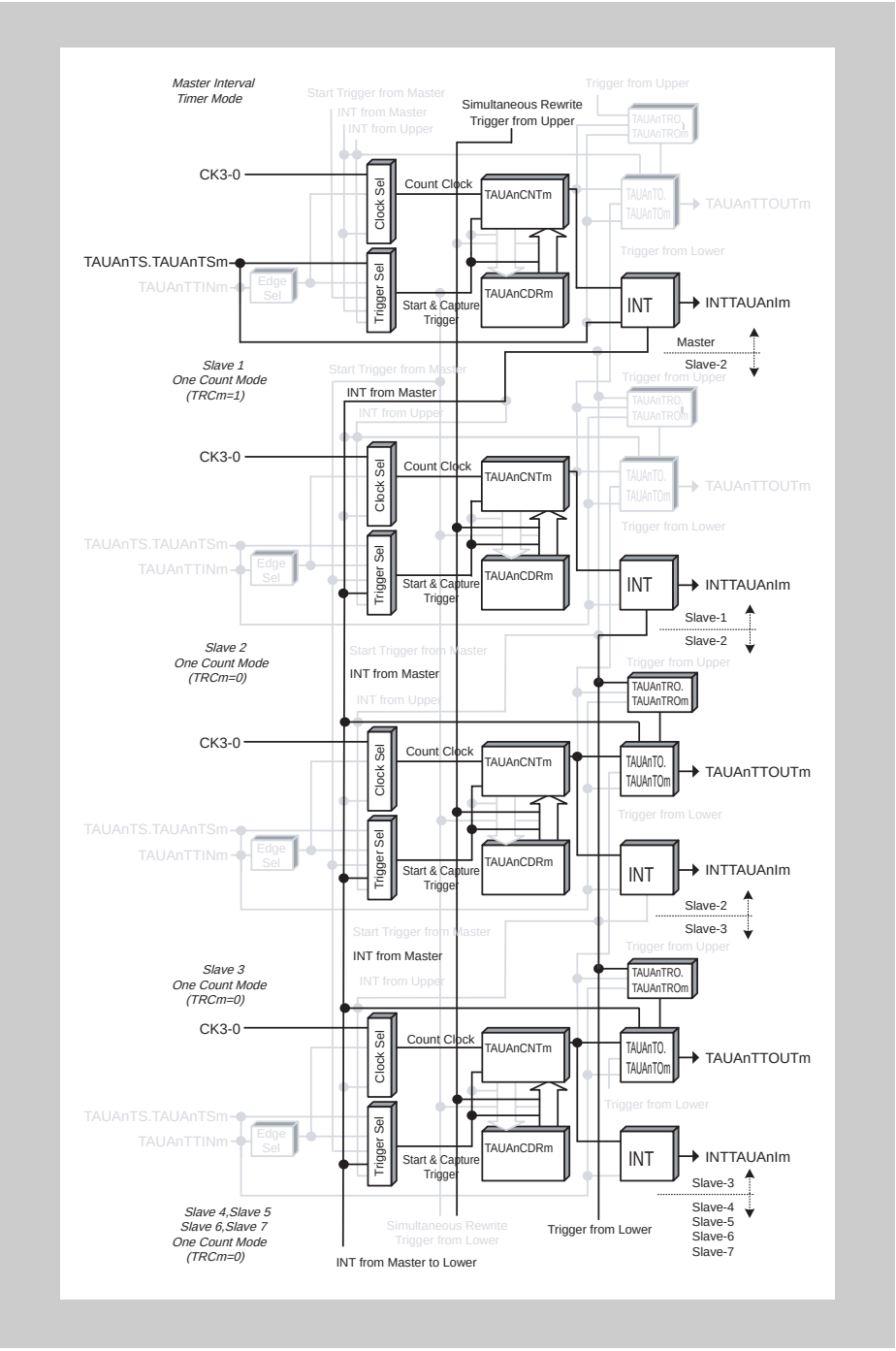


Figure 15-116 Block diagram for non-complementary modulation output function type 1

The following settings apply to the general timing diagram:

- Slave channels 2 to 7: positive logic (TAUAnTOL.TAUAnTOLm = 0)

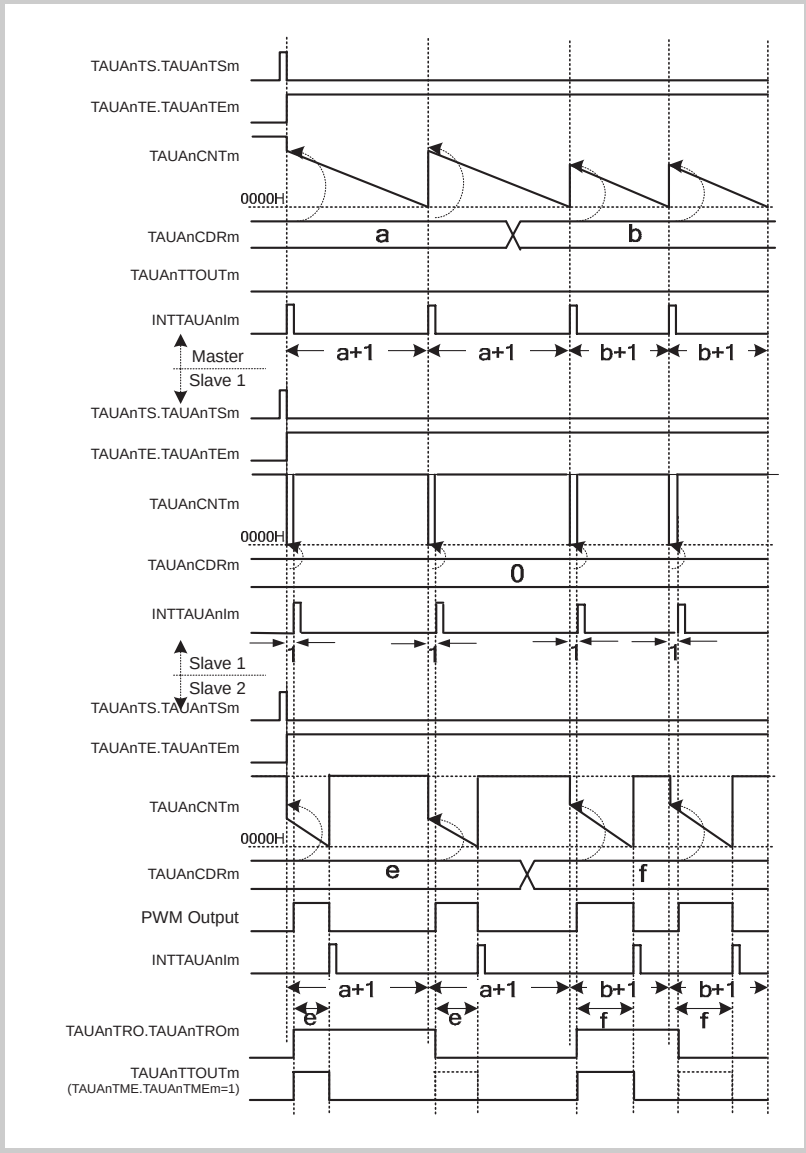


Figure 15-117 General timing diagram for non-complementary modulation output function type 1

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-177 TAUAnCMORm settings for the master channel of non-complementary modulation output function type 1

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channels must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	1: Generates INTTAUAnIm at operation start or restart

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-178 TAUAnCMURm settings for the master channel of non-complementary modulation output function type 1

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the master channel

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-179 Simultaneous rewrite settings for the master channel of non-complementary modulation output function type 1

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

Note If the TAUAnRDS.TAUAnRDSm bit is 1, there must be a channel higher than the master channel to which a simultaneous rewrite trigger signal is generated.

(5) Register settings for slave channel 1

(a) TAUAnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-180 TAUAnCMORm settings for slave channel 1 of non-complementary modulation output function type 1

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channels must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-181 TAUAnCMURm settings for slave channel 1 non-complementary modulation output function type 1

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Because the channel output mode is not used by slave channel 1 of this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used in independent channel output mode controlled by software.

Caution TAUAnTRC.TAUAnTRCm must be set to channel 1 to enable slave 1 to be used as the real-time output trigger.

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-182 Simultaneous rewrite settings for slave channel 1 non-complementary modulation output function type 1

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channels 2 to 7

(a) TAUAnCMORm for slave channels 2 to 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-183 TAUAnCMORm settings for slave channels 2 to 7 of the non-complementary modulation output function type 1

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channels must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm for slave channels 2 to 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-184 TAUAnCMURm settings for slave channels 2 to 7 of the non-complementary modulation output function type 1

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of slave channels 2 to 7

Table 15-185 Control bit settings for slave channels 2 to 7 of the synchronous channel output mode 1 with non-complementary modulation output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	1: Enables real-time output
TAUAnTRO.TAUAnTROm	0: Real-time output is low 1: Real-time output is high
TAUAnTRC.TAUAnTRCm	0: The upper channel generates the real-time trigger for channel m
TAUAnTME.TAUAnTMEm	0: Disables modulation 1: Enables modulation

(d) Simultaneous rewrite for slave channels 2 to 7

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-186 Simultaneous rewrite settings for slave channels 2 to 7 of the non-complementary modulation output function type 1

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for non-complementary modulation output function type 1

Table 15-187 Operating procedure for non-complementary modulation output function type 1 (1/2)

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 858*. Slave channel 1: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for slave channel 1" on page 860*. Slave channels 2 to 7: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(6) "Register settings for slave channels 2 to 7" on page 862*. Set the values of the TAUAnCDRm registers of all channels: set the pulse cycle in TAUAnCDRm of the master channel, set TAUAnCDRm of slave channel 1 to 0000_H, and set the duty width in TAUAnCDRm of slave channels 2 to 7. Set TAUAnTRC.TAUAnTRCm = 1 for slave channel 1.	Channel operation is stopped.

Table 15-187 Operating procedure for non-complementary modulation output function type 1 (2/2)

	Operation	Status of TAUAn
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously (for channel restart, only slaves 2 to 7). TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm of the master and slave channels is set to 1 and the counters start to count down.
During operation	TAUAnCDRm, TAUAnTRO.TAUAnTROm, and TAUAnTME.TAUAnTMEm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	TAUAnCNTm loads the TAUAnCDRm value of the master channel and slave channels 1 to 7, and then counts down. When the counter of the master channel reaches 0000_H: • INTTAUAnIm is generated. • TAUAnCNTm reloads the TAUAnCDRm value, and then continues counting down. • The PWM output signals of slave channels 2 to 7 are set/reset. • TAUAnCNTm reloads the TAUAnCDRm value of slave channel 1, and then counts down. • TAUAnCNTm reloads the TAUAnCDRm value of slave channels 2 to 7, and then counts down. • When the counter of slave channel 1 or slave channels 2 to 7 reaches 0000_H: - INTTAUAnIm is generated. • When the counter of slave channels 2 to 7 reaches 0000_H: - The PWM output signals of slave channels 2 to 7 are set/reset. TAUAnTTOUTm of slave channels 2 to 7 outputs a PWM signal, a high signal, or a low signal depending on the value of the real-time output bits (TAUAnTRO.TAUAnTROm) and the modulation output bits (TAUAnTME.TAUAnTMEm) of a pair of slave channels.
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(8) Specific timing diagrams

The following settings apply to the specific timing diagram:

- Slave channels 2 to 7: positive logic (TAUAnTOL.TAUAnTOLm = 0)

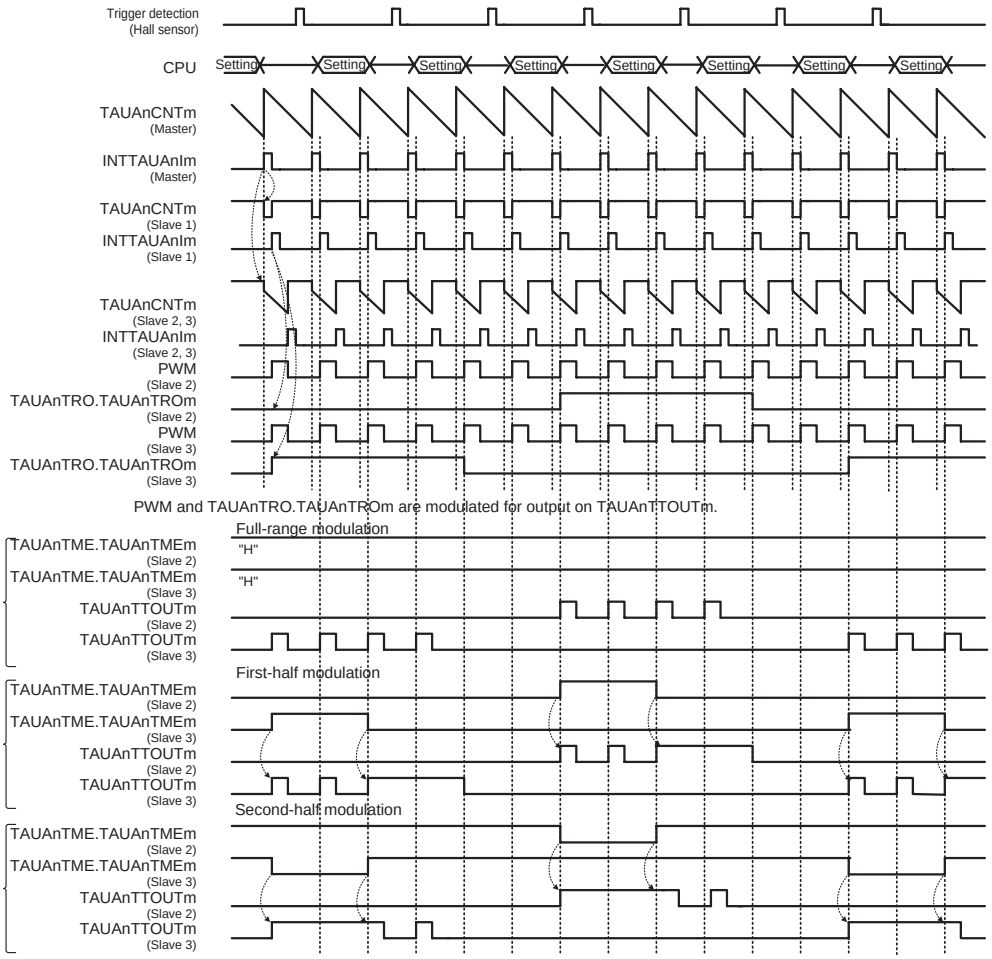


Figure 15-118 Specific timing diagram for non-complementary modulation output function type 1

The above timing diagram shows how full modulation, first-half modulation, and second-half modulation can be achieved by modifying the TAUAnTME.TAUAnTMEm bits of the lower slave channels during operation.

The “Setting” symbol indicates the time period during which the values of TAUAnCDRm, TAUAnTME.TAUAnTMEm and TAUAnTRO.TAUAnTROm can be changed.

The TAUAnTME.TAUAnTMEm setting is applied when counting starts or when a master channel cycle is detected. A modulation waveform is output from TAUAnTTOUTm according to the changed setting.

The values of the TAUAnTRO.TAUAnTROm bits are specified by software, but the new values are only applied when an interrupt occurs on slave channel 1.

15.26.2 Non-complementary modulation output function type 2

(1) Overview

Summary This function outputs a PWM signal, a high signal, or a low signal from TAUAnTTOUTm depending on the value of the real-time output bits (TAUAnTRO.TAUAnTROm) and the modulation output enable bits (TAUAnTME.TAUAnTMEm) of a pair of slave channels. Three pairs of channels are typically used.

Prerequisites

- One master channel and seven slave channels
- The operation mode of the master channel must be set to interval timer mode. See *Table 15-189 "TAUAnCMORm settings for the master channel of non-complementary modulation output function type 2" on page 872*.
- The operation mode of slave channel 1 must be set to event count mode. See *Table 15-193 "TAUAnCMORm settings for slave channel 1 of non-complementary modulation output function type 2" on page 874*.
- The operation mode of slave channels 2 to 7 must be set to up/down count mode. See *Table 15-196 "TAUAnCMORm settings for slave channels 2 to 7 of non-complementary modulation output function type 2" on page 876*.
- TAUAnTTOUTm is not used for the master channel of this function.
- TAUAnTTOUTm of slave channel 1 is not used for this function, but TAUAnTRC.TAUAnTRCm must be set to 1. See *15.9 "Channel Output Modes" on page 627*.
- The channel output mode of slave channels 2 to 7 must be set to Simultaneous Channel Output Mode 2 with non-complementary modulation output. See *15.9 "Channel Output Modes" on page 627*.

Description The counters of the master and slave channels are started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation. The values of the data registers (TAUAnCDRm) are loaded to the counters (TAUAnCNTm).

- Master channel:

  The counter of the master channel starts to count down. When the counter reaches 0000_H, INTTAUAnIm is generated.

- Slave channel 1:

  When slave channel 1 detects an interrupt on the master channel, counter reduces by 1. When the counter reaches 0000_H, it awaits the next interrupt from the master channel. Next, the value of TAUAnCDRm is reloaded to TAUAnCNTm (slave 1), and then INTTAUAnIm is generated.

  Because slave channel 1 is specified as the trigger channel for real-time output (TAUAnTRC.TAUAnTRCm = 1), when an interrupt is generated on slave channel 1, the real-time output bit (TAUAnTRO.TAUAnTROm) value of the channel that detects the generated interrupt of the corresponding channel is changed.

- Slave channel 2:

  When an interrupt is detected from the master channel, the TAUAnCNTm counts in the reverse direction. If the interrupt is detected during up count, the TAUAnCNTm reloads the value of TAUAnCDRm, and then starts counting down.

  When TAUAnCNTm = 0001_H, an interrupt is generated and the PWM output is set or reset.

The combination of the master channel and slave channel 2 generates a PWM output. The master channel generates the period of the PWM output, the slave channel the duty cycle.

Slave channels 3 to 7 behave analogously to slave channel 2.

The signal that is output from TAUAnTTOUTm depends on the value of the real-time output bit (TAUAnTRO.TAUAnTROm) and the modulation output bit (TAUAnTME.TAUAnTMEm) of the slave channel, as shown in *Table 15-188 "TAUAnTTOUTm output of a pair of slave channels in non-complementary modulation output function type 2" on page 868*.

Forced restart is not possible with this function. The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TAUAnTSm to 1.

Conditions

- If TAUAnTME.TAUAnTMEm of the slave channel is 1:
  - If TAUAnTRO.TAUAnTROm of the channel is 1, TAUAnTTOUTm outputs the corresponding PWM of the channel
  - If TAUAnTRO.TAUAnTROm of the channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTME.TAUAnTMEm of the slave channel is 0:
  - If TAUAnTRO.TAUAnTROm of the channel is 1, TAUAnTTOUTm outputs a high signal
  - If TAUAnTRO.TAUAnTROm of the channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTOL.TAUAnTOLm = 1 the high and low signals output from TAUAnTTOUTm are inverted. The PWM signals remain unaffected. Only the initial setting can be used for TAUAnTOL.TAUAnTOLm (and this setting cannot be changed during operation).

Table 15-188 TAUAnTTOUTm output of a pair of slave channels in non-complementary modulation output function type 2

TAUAnTME.TAUAnTMEm	TAUAnTRO.TAUAnTROm	TAUAnTTOUTm outputs
0	0	Low
	1	High
1	0	Low
	1	PWMm

- Simultaneous rewrite can be used with this function. See *15.8 "Simultaneous Rewrite" on page 615*.
- If TAUAnTOL.TAUAnTOLm is cleared to 0 on slave channels 2 to 7, TAUAnTO.TAUAnTOm is set to the low level before clearing TAUAnTE.TAUAnTEm to 0.
- If TAUAnTOL.TAUAnTOLm is set to 1 on slave channels 2 to 7, TAUAnTO.TAUAnTOm is set to the high level before clearing TAUAnTE.TAUAnTEm to 0.

(2) Equations

For slave channels 2 to 7:

PWM output cycle time
= [TAUAnCDRm (master) + 1] x count clock

PWM output duty time
= [TAUAnCDRm (master) + 1 - TAUAnCDRm (slave)] x 2 x count clock

(3) Block diagram and general timing diagram

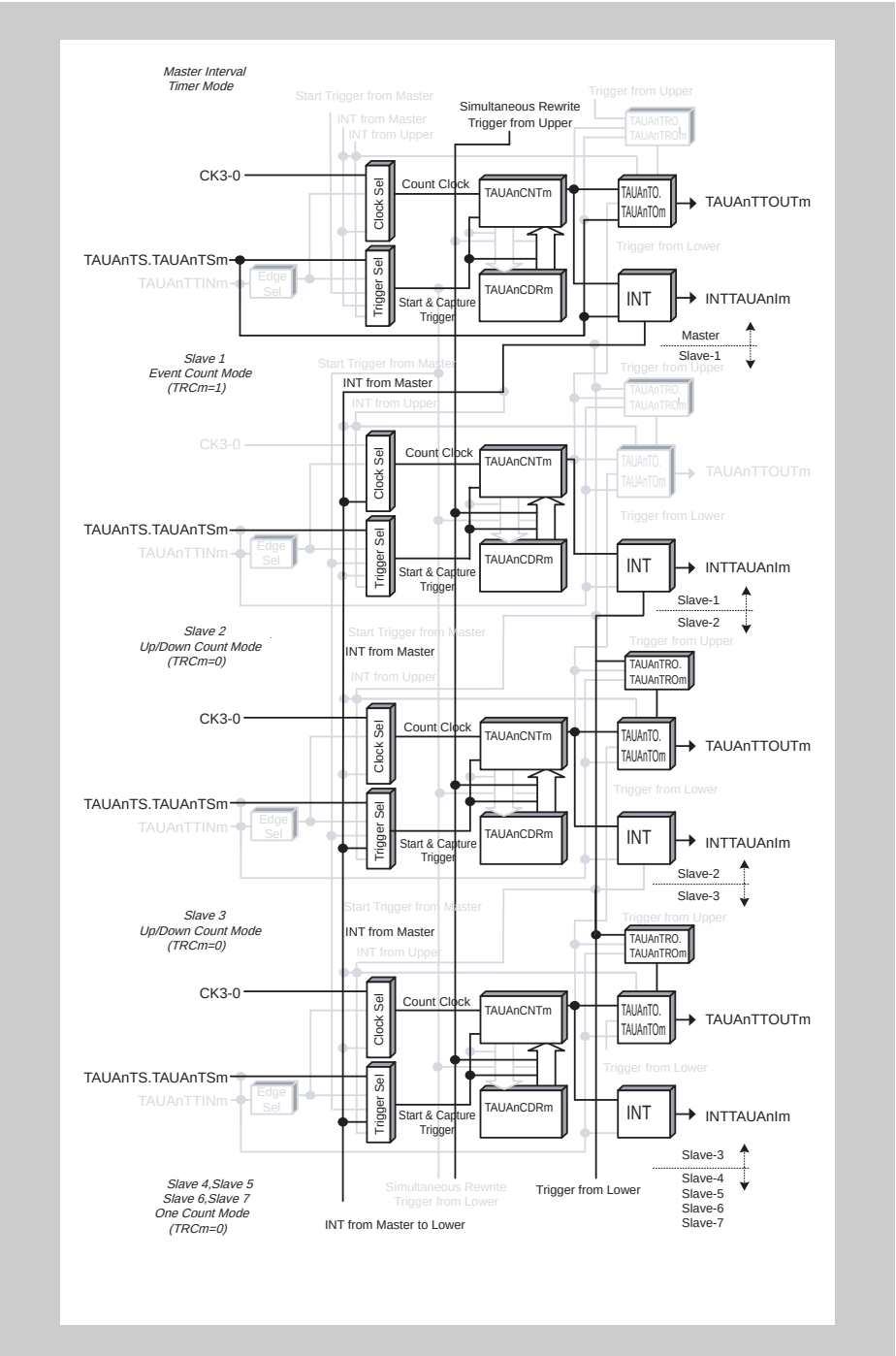


Figure 15-119 Block diagram for non-complementary modulation output function type 2

The following settings apply to the general timing diagram:

- Master channel: INTTAUAnIm not generated at operation start (TAUAnCMORm.TAUAnMD0 = 0)
- Slave channels 2 to 7: positive logic (TAUAnTOL.TAUAnTOLm = 0)

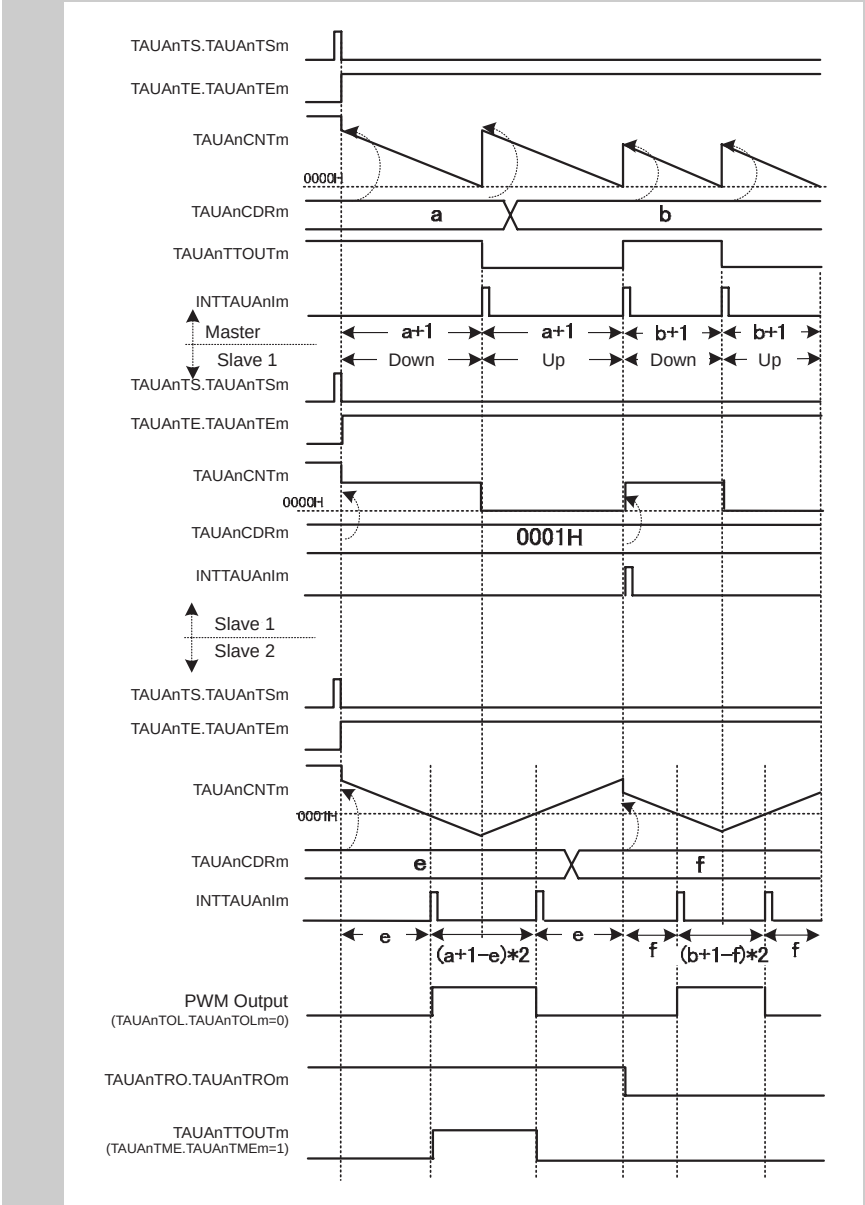


Figure 15-120 General timing diagram for non-complementary modulation output function type 2

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-189 TAUAnCMORm settings for the master channel of non-complementary modulation output function type 2

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channels must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start or restart 1: Generates INTTAUAnIm at operation start or restart

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-190 TAUAnCMURm settings for the master channel of non-complementary modulation output function type 2

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the master channel

Table 15-191 Master channel control bit settings when using non-complementary modulation output function type 2

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables the independent channel output mode.
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (the toggle mode if TAUAnTOM.TAUAnTOMm is 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation.
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), clear these bits to 0.
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output.
TAUAnTRO.TAUAnTROm	0: The real-time output is low.
TAUAnTRC.TAUAnTRCm	0: The next upper channel generates the real-time trigger for channel m.
TAUAnTME.TAUAnTMEm	0: Disables modulation.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-192 Simultaneous rewrite settings for the master channel of non-complementary modulation output function type 2

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: When simultaneous rewrite is disabled (TAUAnTAUAnRDE.TAUAnRDEm = 0), set these bits to 0. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

Note If the TAUAnRDS.TAUAnRDSm bit is 1, there must be a channel higher than the master channel to which a simultaneous rewrite trigger signal is generated.

(5) Register settings for slave channel 1

(a) TAUAnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-193 TAUAnCMORm settings for slave channel 1 of non-complementary modulation output function type 2

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channels must be identical.
TAUAnCCS[1:0]	11: INTTAUAnIm of the master channel is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger 011: Simultaneous rewrite trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0011: Event count mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start or restart

(b) TAUAnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-194 TAUAnCMURm settings for slave channel 1 of non-complementary modulation output function type 2

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Because the channel output mode is not used by slave channel 1 of this function, clear TAUAnTOEn.TAUAnTOEn. However, it can be used in independent channel output mode controlled by software.

Caution TAUAnTRC.TAUAnTRCm must be set to 1 to enable slave channel 1 to be used as the real-time output trigger.

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-195 Simultaneous rewrite settings for slave channel 1 of non-complementary modulation output function type 2

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channels 2 to 7

(a) TAUAnCMORm for slave channels 2 to 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-196 TAUAnCMORm settings for slave channels 2 to 7 of non-complementary modulation output function type 2

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channels must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	111: The up/down output trigger signal TAUAnTUDSm of the master channel is the start trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	1001: Up/down count mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start or restart

(b) TAUAnCMURm for slave channels 2 to 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-197 TAUAnCMURm settings slave channels 2 to 7 of non-complementary modulation output function type 2

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of slave channels 2 to 7

Table 15-198 Control bit settings slave channels 2 to 7 of synchronous channel output mode 2 with non-complementary modulation output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel output
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	1: Enables real-time output
TAUAnTRO.TAUAnTROm	0: Real-time output is low 1: Real-time output is high
TAUAnTRC.TAUAnTRCm	0: The next upper channel generates the real-time trigger for channel m
TAUAnTME.TAUAnTMEm	0: Disables modulation 1: Enables modulation

(d) Simultaneous rewrite for slave channels 2 to 7

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-199 Simultaneous rewrite settings slave channels 2 to 7 of non-complementary modulation output function type 2

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for non-complementary modulation output function type 2

Table 15-200 Operating procedure for non-complementary modulation output function type 2 (1/2)

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 872*. Slave channel 1: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for slave channel 1" on page 874*. Slave channels 2 to 7: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(6) "Register settings for slave channels 2 to 7" on page 876*. Set the values of the TAUAnCDRm registers of all channels: set the pulse cycle in TAUAnCDRm of the master channel, using TAUAnCDRm of slave channel 1, set the number of interrupts on the master channel to be ignored before slave 1 generates an input, and set the duty width in TAUAnCDRm of slave channels 2 to 7. Set TAUAnTRC.TAUAnTRCm = 1 for slave channel 1.	Channel operation is stopped.

Table 15-200 Operating procedure for non-complementary modulation output function type 2 (2/2)

	Operation	Status of TAUAn
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously (for channel restart, only slaves 2 to 7). TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm of the master and slave channels is set to 1 and the counters start to count down.
During operation	TAUAnCDRm, TAUAnTRO.TAUAnTROm, and TAUAnTME.TAUAnTMEm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	TAUAnCNTm loads the TAUAnCDRm value of the master channel and slave channels 2 and 7, and then counts down. The TAUAnCDRm value of slave channel 1 is loaded, and a master channel interrupt is awaited. When the counter of the master channel reaches 0000_H: • INTTAUAnIm is generated. • TAUAnCNTm reloads the TAUAnCDRm value, and then continues counting down. • TAUAnCNTm of slave channel 1 reduces by 1 and awaits the next interrupt on the master channel. • TAUAnCNTm of slave channels 2 to 7 counts in the other direction. • When the counter of slave channel 1 reaches 0000_H it awaits the next interrupt from the master channel. When it is detected: - INTTAUAnIm is generated. • When the counter of slave channels 2 to 7 reaches 0001_H: - INTTAUAnIm is generated. - The PWM output signals of slave channels 2 to 7 are set/reset. TAUAnTTOUTm of slave channels 2 to 7 outputs a PWM signal, a high signal, or a low signal depending on the value of the real-time output bits (TAUAnTRO.TAUAnTROm) and the modulation output bits (TAUAnTME.TAUAnTMEm) of a pair of slave channels.
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(8) Specific timing diagrams

The following settings apply to the general timing diagram:

- Slave channels 2 to 7: positive logic (TAUAnTOL.TAUAnTOLm = 0)

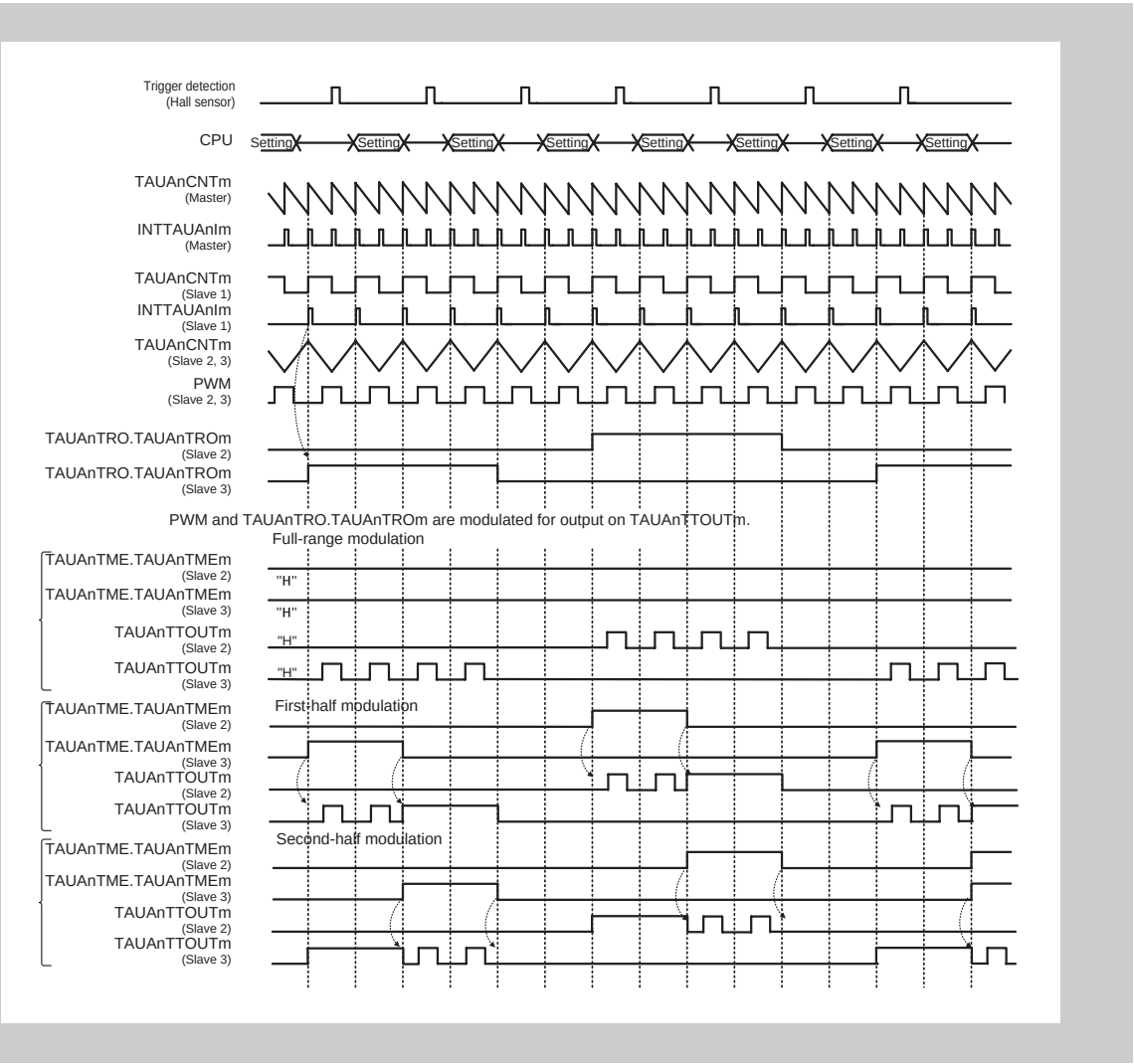


Figure 15-121 Specific timing diagram for non-complementary modulation output function type 2

The above timing diagram shows how full modulation, first-half modulation, and second-half modulation can be achieved by modifying the TAUAnTME.TAUAnTMEm bits of the lower slave channels during operation.

The “Setting” symbol indicates the time period during which the values of TAUAnCDRm, TAUAnTME.TAUAnTMEm and TAUAnTRO.TAUAnTROm can be changed.

The TAUAnTME.TAUAnTMEm setting is applied when counting starts or when a triangle PWM carrier cycle (at the peak interrupt timing) is detected.

The values of the TAUAnTRO.TAUAnTROm bits are specified by software, but the new values are only applied when an interrupt occurs on slave channel 1.

15.26.3 Complementary modulation output function

(1) Overview

Summary This function outputs a PWM signal, a high signal, or a low signal from TAUAnTTOUTm depending on the value of the real-time output bits (TAUAnTRO.TAUAnTROm), the modulation output bits (TAUAnTME.TAUAnTMEm), and the output level bits (TAUAnTDL.TAUAnTDLm) of a pair of slave channels. Three pairs of channels are typically used.

Prerequisites

- One master channel and seven slave channels
- The operation mode of the master channel must be set to interval timer mode. See *Table 15-203 "TAUAnCMORm settings for the master channel of the complementary modulation output function" on page 887*.
- The operation mode of slave channel 1 must be set to event count mode. See *Table 15-207 "TAUAnCMORm settings for slave channel 1 of the complementary modulation output function" on page 889*.
- The operation mode of slave channels 2, 4, and 6 must be set to up/down count mode. See *Table 15-210 "TAUAnCMORm settings for slave channels 2, 4, and 6 of the complementary modulation output function" on page 891*.
- The operation mode of slave channels 3, 5, and 7 must be set to one-count mode. See *Table 15-214 "TAUAnCMORm settings for slave channels 3, 5, and 7 of the complementary modulation output function" on page 893*.
- The channel output mode of the master channel must be set to independent channel output mode 1. See *15.9 "Channel Output Modes" on page 627*.
- TAUAnTTOUTm of slave channel 1 is not used for this function, but TAUAnTRC.TAUAnTRCm must be set to 1. See *15.9 "Channel Output Modes" on page 627*.
- The channel output mode of slave channels 2 to 7 must be set to Simultaneous Channel Output Mode 2 with complementary modulation output. See *15.9 "Channel Output Modes" on page 627*.

Description

- Master channel:

  The counter of the master channel is started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation. The value of the data register of the master channel (TAUAnCDRm) is loaded to the counter (TAUAnCNTm) and the counter starts to count down from this value.

  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated. This triggers the counter of slave channel 1 to reduce by 1 and the counter of slave channel 2 to count in the reverse direction.

- Slave channel 1:

  When the counter reaches 0000_H, it awaits the next interrupt from the master channel. Next, TAUAnCNTm (slave 1) reloads the value of INTTAUAnIm, and then INTTAUAnIm is generated.

  Slave channel 1 is set as the channel that triggers real-time output (TAUAnTRC.TAUAnTRCm = 1). For each channel on which an interrupt generated on slave channel 1 is detected, the real-time output bit (TAUAnTRO.TAUAnTROm) value is applied. The value of the real-time output bits can be changed at any time by the application software, but the new values are only applied when the interrupt occurs on slave channel 1.

- Slave channel 2:

  When the counter of slave channel 2 reaches 0001_H, the counter of slave channel 3 starts to count down. When the counter of slave channel 3 reaches 0000_H it generates an interrupt.

- Slave channels 2 and 3:

  The combination of the master channel and slave channels 2 and 3 generates a PWM output. The master channel generates the period of the PWM output, slave channel 2 the duty cycle, and slave channel 3 generates the dead time.

- Slave channels 4 to 7:

  Slave channels 4 and 6 behave analogously to slave channel 2. Slave channels 5 and 7 behave analogously to slave channel 3.

The signal that is output from TAUAnTTOUTm depends on the value of the real-time output bits (TAUAnTRO.TAUAnTROm), the modulation output bits (TAUAnTME.TAUAnTMEm), and the output level bits (TAUAnTOL.TAUAnTOLm) of a pair of slave channels, as shown in *Table 15-176 "TAUAnTTOUTm output of a pair of slave channels in non-complementary modulation output function type 1" on page 855*.

However, the output of slave channels 2 and 3 must not be high simultaneously (for example to prevent a motor driver from short circuiting). To prevent both channels being high at the instant that one channel switches to low and the other switches to high, a delay must be added to the channel that switches to high so that this transition occurs later. This is achieved by adding dead time to either the positive or negative-phased PWM signal as specified by the value of the channel dead time output level register (TAUAnTDL.TAUAnTDLm).

The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TAUAnTSm to 1.

Conditions

- If TAUAnTME.TAUAnTMEm of both channels in a pair is 1:
  - If the TAUAnTRO.TAUAnTROm value of one channel is 1, TAUAnTTOUTm performs PWM output for that channel.
  - If the TAUAnTRO.TAUAnTROm value of one channel is 0, TAUAnTTOUTm outputs a low level signal.
- If TAUAnTME.TAUAnTMEm of both channels in a pair is 0:
  - If TAUAnTRO.TAUAnTROm of a channel is 1, TAUAnTTOUTm outputs a high signal
  - If TAUAnTRO.TAUAnTROm of a channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTOL.TAUAnTOLm = 1, the high and low signals output from TAUAnTTOUTm are inverted. The PWM signal does not change in accordance with the TAUAnTOL.TAUAnTOLm setting.

Table 15-201 TAUAnTTOUTm output of a pair of slave channels in complementary modulation output function for TAUAnTOL.TAUAnTOLm = 0

TAUAnTME.TAUAnTME2	TAUAnTME.TAUAnTME3	TAUAnTRO.TAUAnTRO2	TAUAnTRO.TAUAnTRO3	TAUAnTTOUT2 outputs	TAUAnTTOUT3 outputs
0	0	0	0	LOW	LOW
		0	1	LOW	HIGH
		1	0	HIGH	LOW
		1	1	Prohibited	Prohibited
1	1	0	0	LOW	LOW
		0	1	~PWMm	PWMm
		1	0	PWMm	~PWMm
		1	1	Prohibited	Prohibited

Notes 1. ~PWM indicates an inverted PWM signal. PWM and -PWM are set up using TAUAnTDL.TAUAnTDLm.

2. All settings that are not listed in this table are not permitted.

- While TAUAnTRO.TAUAnTROm is set to 1 on one set of channels, if TAUAnTME.TAUAnTMEm is continuously set to 1, the modulation is entire modulation.
- If TAUAnTME.TAUAnTMEm is set to 1 during the first half of the period during which TAUAnTRO.TAUAnTROm is set to 1 on one set of channels, the modulation is first-half modulation.
- If TAUAnTME.TAUAnTMEm is set to 1 during the second half of the period during which TAUAnTRO.TAUAnTROm is set to 1 on one set of channels, the modulation is second-half modulation.
- If two channels simultaneously output a high level signal, the TAUAnTDL.TAUAnTDLm bit value determines whether the dead time is added to the positive-phase PWM signal or the negative-phase PWM signal.
  - If TAUAnTDL.TAUAnTDLm = 0, dead time is added to the positive PWM signal
  - If TAUAnTDL.TAUAnTDLm = 1 dead time is added to the negative PWM signal
  - The values of the TAUAnTDL.TAUAnTDLm bits must be manipulated by the application software during operation. To change TAUAnTDL.TAUAnTDLm, perform rewriting while TAUAnTRO.TAUAnTROm is 00B.

Table 15-202 TAUAnTDL.TAUAnTDLm settings for a pair of slave channels in complementary modulation output function for TAUAnTOL.TAUAnTOLm = 0

TAUAnTME. TAUAnTME2	TAUAnTME. TAUAnTME3	TAUAnTRO. TAUAnTRO2	TAUAnTRO. TAUAnTRO3	TAUAnTDL. TAUAnTDL2	TAUAnTDL. TAUAnTDL3
0	0	0	0	1	1
		0	1	1	0
		1	0	0	1
1	1	0	0	1	1
		0	1	1	0
		1	0	0	1

- The value of TAUAnCDRm of slave channel 1 must be set so that INTTAUAnIm is generated from slave 1 at the summit of a carrier cycle.
- Clear the TAUAnCMORm.TAUAnMD0 value of the master channel to 0.
- Simultaneous rewrite can be used with this function. See *15.8 "Simultaneous Rewrite" on page 615*.

(2) Equations

For slaves 2, 4, and 6

When TAUAnTOL.TAUAnTOLm = 0 and TAUAnCDR (slave 1):

PWM output cycle time = 2 x [TAUAnCDRm (master) + 1] x count clock

PWM output duty time = { [TAUAnCDRm (master) + 1 - TAUAnCDRm (slave 2)] x 2 - [TAUAnCDRm (slave 3) + 1] } x count clock

(3) Block diagram and general timing diagram

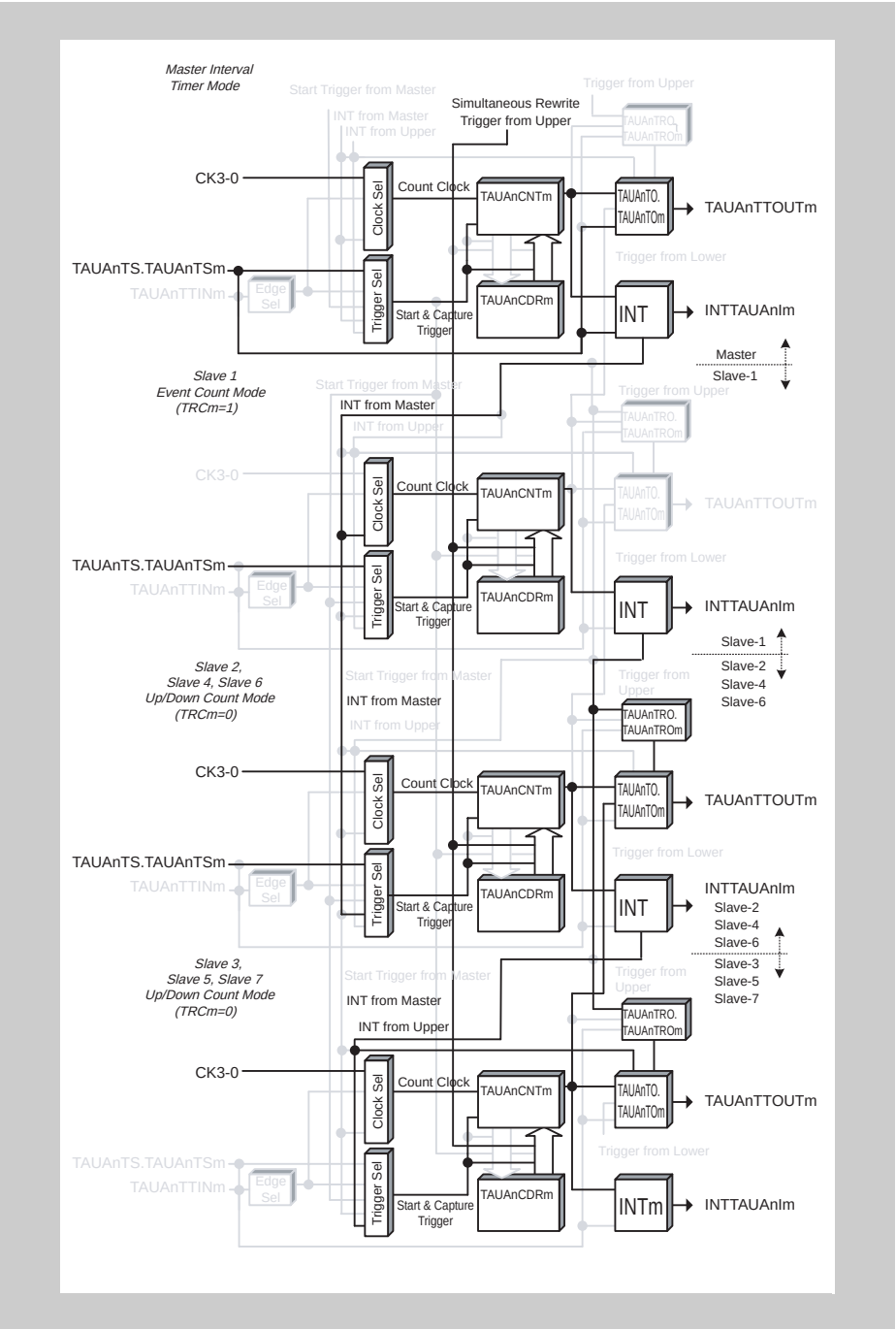


Figure 15-122 Block diagram for complementary modulation output function

The following settings apply to the general timing diagram:

- Master channel: INTTAUAnIm not generated at operation start (TAUAnCMORm.TAUAnMD0 = 0)
- Slave channels 2 to 7: positive logic (TAUAnTOL.TAUAnTOLm = 0)

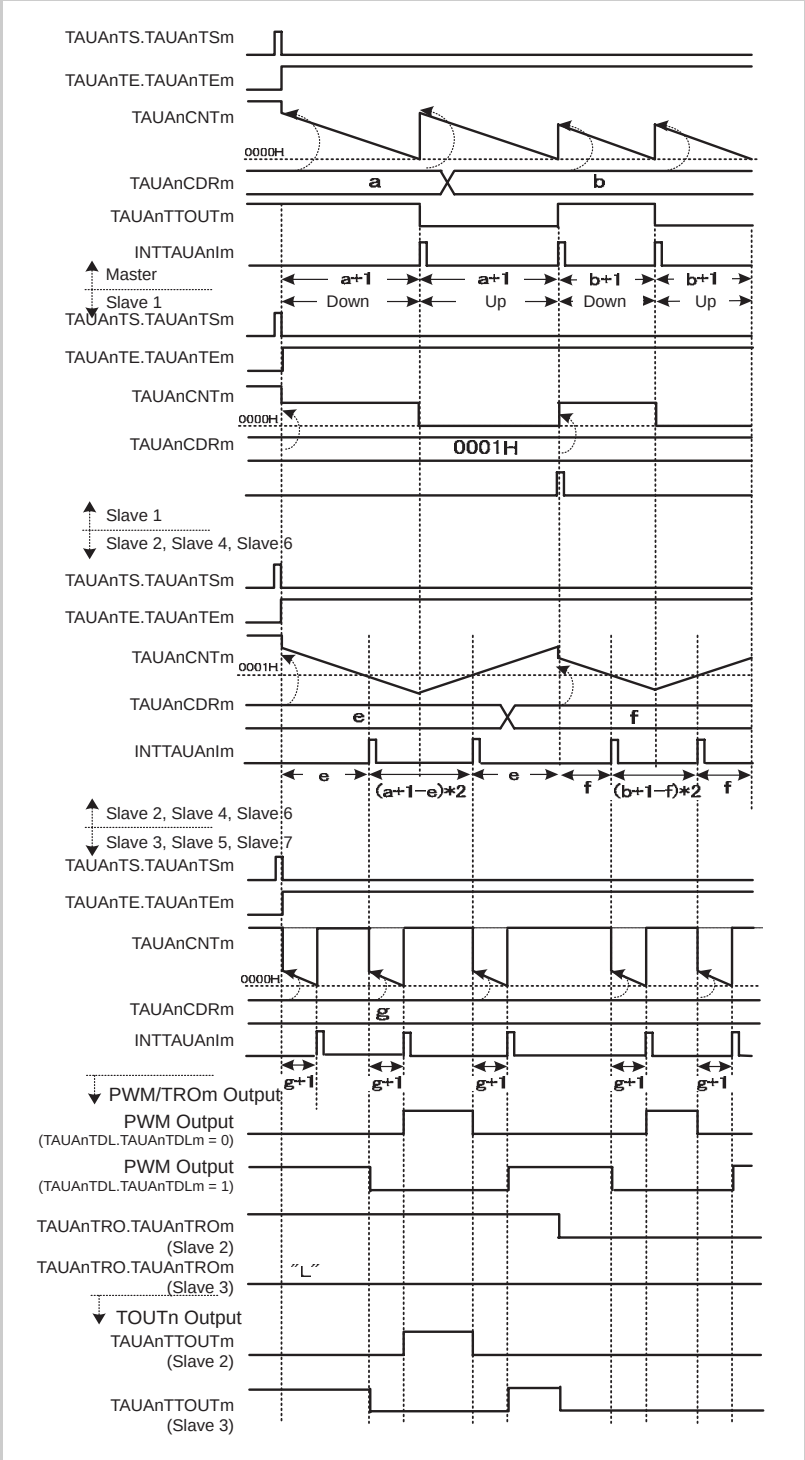


Figure 15-123 General timing diagram for complementary modulation output function

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-203 TAUAnCMORm settings for the master channel of the complementary modulation output function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channels must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start or restart 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-204 TAUAnCMURm settings for the master channel of the complementary modulation output function

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Table 15-205 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	0: Independent channel output
TAUAnTOC.TAUAnTOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TAUAnTOMm = 0)
TAUAnTOL.TAUAnTOLm	0: Positive logic
TAUAnTDE.TAUAnTDEm	0: Disables dead time operation
TAUAnTDM.TAUAnTDMm	0: When dead time operation is disabled (TAUAnTDE.TAUAnTDEm = 0), set these bits to 0
TAUAnTDL.TAUAnTDLm	
TAUAnTRE.TAUAnTREm	0: Disables real-time output
TAUAnTRO.TAUAnTROm	0: When real-time output is disabled (TAUAnTRE.TAUAnTREm = 0), set these bits to 0
TAUAnTRC.TAUAnTRCm	
TAUAnTME.TAUAnTMEm	0: Disables modulation

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-206 Simultaneous rewrite settings for the master channel of the complementary modulation output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

Note If the TAUAnRDS.TAUAnRDSm bit is 1, there must be a channel higher than the master channel to which a simultaneous rewrite trigger signal is generated.

(5) Register settings for slave channel 1

(a) TAUAnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-207 TAUAnCMORm settings for slave channel 1 of the complementary modulation output function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channels must be identical.
TAUAnCCS[1:0]	11: INTTAUAnIm of the master channel is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger 011: Simultaneous rewrite trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0011: Event count mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start or restart

(b) TAUAnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-208 TAUAnCMURm settings for slave channel 1 of the complementary modulation output function

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Because the channel output mode is not used by slave channel 1 of this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used in independent channel output mode controlled by software.

Caution TAUAnTRC.TAUAnTRCm must be set to 1 to enable slave 1 to be used as the real-time output trigger.

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-209 Simultaneous rewrite settings for slave channel 1 of the complementary modulation output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channels 2, 4, and 6

(a) TAUAnCMORm for slave channels 2, 4, and 6

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-210 TAUAnCMORm settings for slave channels 2, 4, and 6 of the complementary modulation output function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channels must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	111: The up/down output trigger signal of the master channel
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	1001: Up/down count mode
TAUAnMD0	0: INTTAUAnIm not generated at operation start or restart

(b) TAUAnCMURm for slave channels 2, 4, and 6

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-211 TAUAnCMURm settings for slave channels 2, 4, and 6 of the complementary modulation output function

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of slave channels 2, 4, and 6

Table 15-212 Control bit settings for synchronous channel output mode 2 with complementary modulation output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel output
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	1: Enables dead time operation
TAUAnTDM.TAUAnTDMm	0: Dead time is added upon detection of a duty cycle at the upper even channel
TAUAnTDL.TAUAnTDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase
TAUAnTRE.TAUAnTREm	1: Enables real-time output
TAUAnTRO.TAUAnTROm	0: Real-time output is low 1: Real-time output is high
TAUAnTRC.TAUAnTRCm	0: The upper channel generates the real-time trigger for channel m
TAUAnTME.TAUAnTMEm	0: Disables modulation 1: Enables modulation

Caution For TAUAnTDL.TAUAnTDLm, specify the setting that is opposite that of the odd channel.

(d) Simultaneous rewrite for slave channels 2, 4, and 6

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-213 Simultaneous rewrite settings for slave channels 2, 4, and 6 of the complementary modulation output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Register settings for slave channels 3, 5, and 7

(a) TAUAnCMORm for slave channels 3, 5, and 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-214 TAUAnCMORm settings for slave channels 3, 5, and 7 of the complementary modulation output function

Bit name	Setting
TAUAnCKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	110: Dead time trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0100: One-count mode
TAUAnMD0	1: Enables start trigger detection during counting

(b) TAUAnCMURm for slave channels 3, 5, and 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-215 TAUAnCMURm settings for slave channels 3, 5, and 7 of the complementary modulation output function

Bit name	Setting
TAUAnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of slave channels 3, 5, and 7

Table 15-216 Control bit settings for synchronous channel output mode 2 with complementary modulation output

Bit name	Setting
TAUAnTOE.TAUAnTOEm	1: Enables independent channel output mode
TAUAnTOM.TAUAnTOMm	1: Synchronous channel output
TAUAnTOC.TAUAnTOCm	1: Operation mode 2
TAUAnTOL.TAUAnTOLm	0: Positive logic 1: Inverted logic
TAUAnTDE.TAUAnTDEm	1: Enables dead time operation
TAUAnTDM.TAUAnTDMm	0: Dead time is added upon detection of a duty cycle at the upper even channel
TAUAnTDL.TAUAnTDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase
TAUAnTRE.TAUAnTREm	1: Enables real-time output
TAUAnTRO.TAUAnTROm	0: Real-time output is low 1: Real-time output is high
TAUAnTRC.TAUAnTRCm	0: The upper channel generates the real-time trigger for channel m
TAUAnTME.TAUAnTMEm	0: Disables modulation 1: Enables modulation

Caution For TAUAnTDL.TAUAnTDLm, specify the setting that is opposite that of the even channel.

(d) Simultaneous rewrite for slave channels 3, 5, and 7

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-217 Simultaneous rewrite settings for slave channels 3, 5, and 7 of the complementary modulation output function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(8) Operating procedure for complementary modulation output function

Table 15-218 Operating procedure for complementary modulation output Function (1/2)

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 887*. Slave channel 1: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for slave channel 1" on page 889*. Slave channels 2, 4, and 6: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(6) "Register settings for slave channels 2, 4, and 6" on page 891*. Slave channels 3, 5, and 7: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(7) "Register settings for slave channels 3, 5, and 7" on page 893*. Set the values of the TAUAnCDRm registers of all channels: set the pulse cycle in TAUAnCDRm of the master channel, set the number of interrupts on the master channel to be ignored using TAUAnCDRm of slave channel 1, and set the duty width in TAUAnCDRm of slave channels 2, 4, and 6, and set the dead time delay in slave channels 3, 5, and 7. Set TAUAnTRC.TAUAnTRCm = 1 for slave channel 1.	Channel operation is stopped.

Table 15-218 Operating procedure for complementary modulation output Function (2/2)

		Operation	Status of TAUAn
Restart	Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously (for channel restart, only slaves 2 to 7). TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm of the master and slave channels is set to 1 and the counters start to count down.
	During operation	TAUAnCDRm, TAUAnTRO.TAUAnTROm, and TAUAnTME.TAUAnTMEm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	TAUAnCNTm loads the TAUAnCDRm value of the master channel and slave channels 2 and 7, and then counts down. The TAUAnCDRm value of slave channel 1 is loaded, and a master channel interrupt is awaited. When the counter of the master channel reaches 0000_H: • INTTAUAnIm is generated. • TAUAnCNTm reloads the TAUAnCDRm value, and then continues counting down. • TAUAnCNTm of slave channel 1 reduces by 1 and awaits the next interrupt on the master channel. • TAUAnCNTm of slave channels 2, 4 and 6 counts in the reverse direction. • When the counter of slave channel 1 reaches. 0000_H it awaits the next interrupt from the master channel. When it is detected: - TAUAnCNTm reloads the TAUAnCDRm value and awaits the next interrupt on the master channel. - INTTAUAnIm is generated. - TAUAnTRO.TAUAnTROm can change. • When the counter of slave channels 2, 4, and 6 reaches 0001_H: - INTTAUAnIm is generated. - The PWM output of slave channel m resets. - TAUAnCNTm loads the TAUAnCDRm value of slave channels 3, 5, and 7, and then counts down. • When the counter of slave channels 3, 5, and 7 reaches 0001_H: - INTTAUAnIm is generated. TAUAnTTOUTm of slave channels 2 to 7 outputs a PWM signal, a high signal, or a low signal depending on the value of the real-time output bits (TAUAnTRO.TAUAnTROm), the modulation output bits (TAUAnTME.TAUAnTMEm), and the output level bits (TAUAnTOL.TAUAnTOLm) of a pair of slave channels.
	Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

(9) Specific timing diagrams

The following settings apply to the timing diagram:

- Slave channels 2 to 7: positive logic (TAUAnTOL.TAUAnTOLm = 0)

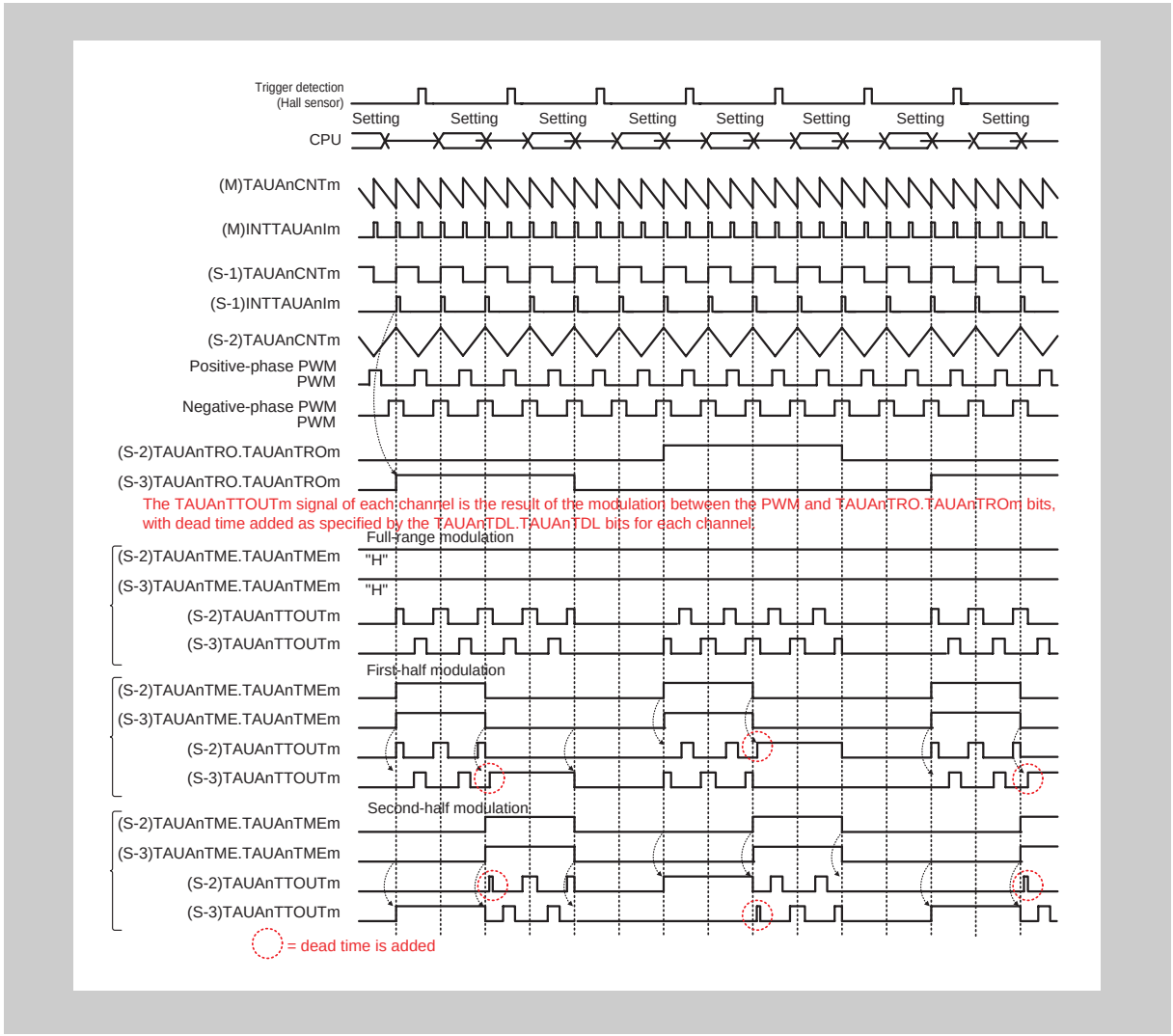


Figure 15-124 Specific timing diagram for complementary modulation output function

The above timing diagram shows how full modulation, first-half modulation, and second-half modulation can be achieved by modifying the TAUAnTME.TAUAnTMEm bits of the lower slave channels during operation.

The output of channels 2 and 3 is the modulation of the PWM output and the value of the TAUAnTRO.TAUAnTROm bits. Thus the type of PWM signal (positive or negative) output by an individual channel varies depending on the value of these bits.

The TAUAnTRO.TAUAnTROm, TAUAnTME.TAUAnTMEm, and TAUAnTDL.TAUAnTDLm bit values are specified using software, but a newly specified value is not applied unless an interrupt is generated on slave channel 1.

Note Dead time is added to prevent the positive- and negative-phase PWM edges from changing simultaneously.

The “Setting” symbol indicates the time period during which the values of TAUAnCDRm, TAUAnTME.TAUAnTMEm, TAUAnTRO.TAUAnTROm, and TAUAnTDL.TAUAnTDLm can be changed.

15.27 Other Synchronous Channel Functions

This chapter describes a function that is used to ignore a specified number of interrupts on the master channel.

15.27.1 Interrupt culling function

(1) Overview

Summary This function divides the number of interrupts of the master channel by a specified value using a slave channel.

The interrupt culling function is a sub function of the following functions:

- PWM output function

  (*15.23.1 "PWM output function" on page 770*)

- Triangle PWM output function

  (*15.25.1 "Triangle PWM output function" on page 826*)

- Triangle PWM output function with dead time

  (*15.25.2 "Triangle PWM output function with dead time" on page 837*)

Prerequisites

- Two channels
- The operation mode of the master channel must be set to interval timer mode. See *Table 15-219 "TAUAnCMORm settings for the master channel of the interrupt culling function" on page 903*.
- The operation mode of the slave channel must be set to event count mode. See *Table 15-222 "TAUAnCMORm settings for the slave channel of the interrupt culling function" on page 905*.
- TAUAnTTOUTm is not used for the slave channel of this function

Description The counters (master and slave) are started by setting the channel trigger bit (TAUAnTS.TAUAnTSm) to 1 for both channels. This in turn sets TAUAnTE.TAUAnTEm, enabling count operation. The current value of the data register of the master channel and slave channel (TAUAnCDRm) are loaded to the counter (TAUAnCNTm).

- Master channel:

  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated and TAUAnCNTm loads the TAUAnCDRm value.

- Slave channel:

  Every time the master channel generates an INTTAUAnIm, the counter of the slave channel reduces by one. When the counter reaches 0000_H, it awaits the next interrupt from the master channel. This causes TAUAnCNTm (slave) to load the value of TAUAnCDRm, and an INTTAUAnIm is generated.

Forced restart is not possible for this function. The counter can be stopped by setting TAUAnTT.TAUAnTTm to 1 for the master and slave channel, which in turn sets TAUAnTE.TAUAnTEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel stop but retain their values.

Conditions Simultaneous rewrite can be used with this function. See *15.8 "Simultaneous Rewrite" on page 615*.

(2) Equations

Interrupt divider = TAUAnCDRm (slave channel)

– One INTTAUAnIm is generated for the number of INTTAUAnIm of the master channel defined as TAUAnCDRm (slave channel) + 1.

(3) Block diagram and general timing diagram

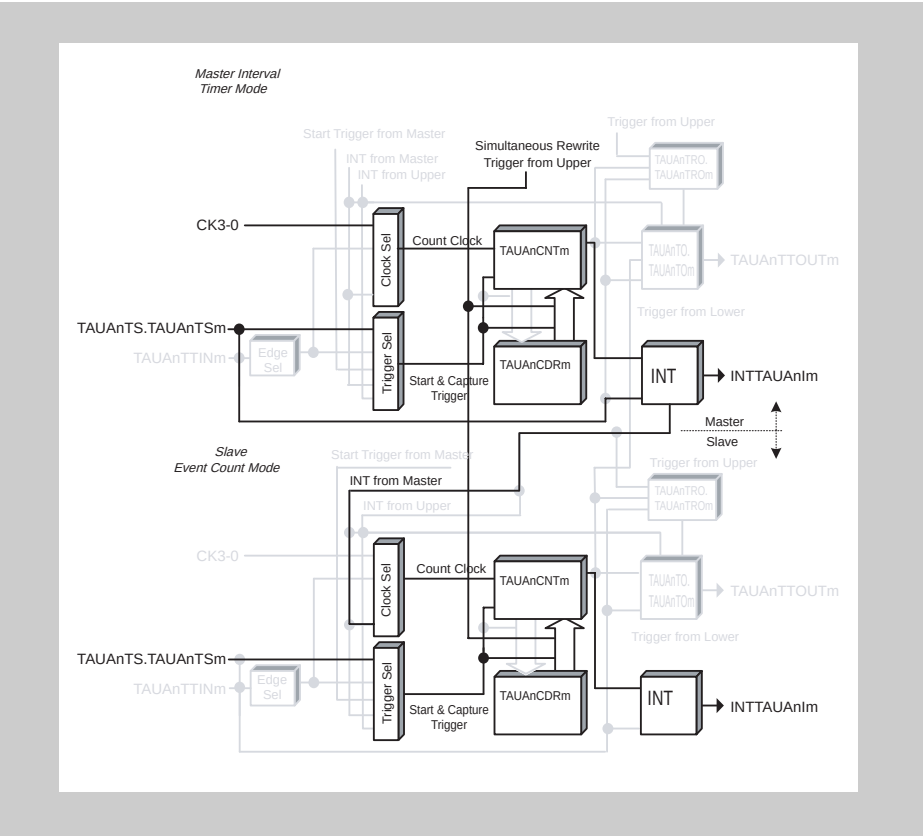


Figure 15-125 Block diagram for interrupt culling function

The following settings apply to the general timing diagram:

Master channel:

- INTTAUAnIm is generated at operation start (TAUAnCMORm.TAUAnMD0 = 1)

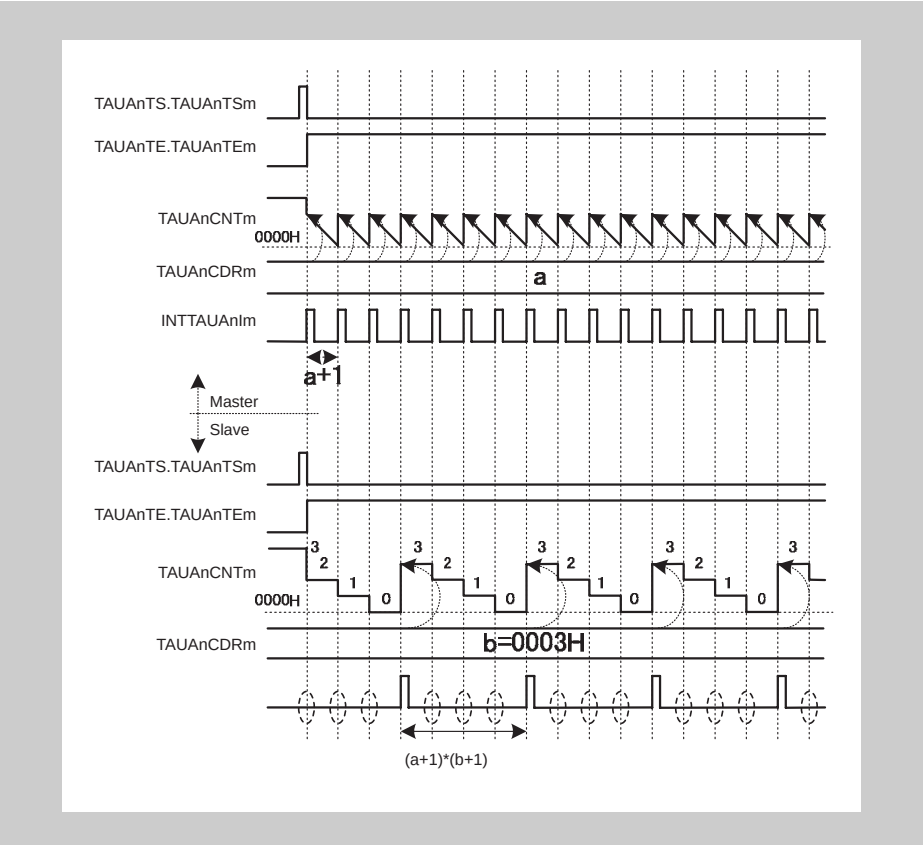


Figure 15-126 General timing diagram for interrupt culling function

The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave.

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-219 TAUAnCMORm settings for the master channel of the interrupt culling function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	00: Operation clock is used as the count clock
TAUAnMAS	1: Channel is master channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0000: Interval timer mode
TAUAnMD0	0: INTTAUAnIm not output at operation startup 1: Generates INTTAUAnIm at operation startup

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-220 TAUAnCMURm settings for the master channel of the interrupt culling function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the master channel

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEn. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-221 Simultaneous rewrite settings for the master channel of the interrupt culling function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting 1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel

(a) TAUAnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:1]				TAUAn MD0

Table 15-222 TAUAnCMORm settings for the slave channel of the interrupt culling function

Bit name	Setting
TAUAnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUAnCKS[1:0] bit of the master and slave channel must be identical.
TAUAnCCS[1:0]	11: INTTAUAnIm of the master channel is used as the count clock
TAUAnMAS	0: Channel is a slave channel
TAUAnSTS[2:0]	000: Counter triggered by software trigger
TAUAnCOS[1:0]	00: Not used, so set to 00
TAUAnMD[4:1]	0011: Event count mode
TAUAnMD0	0: INTTAUAnIm not output at operation startup

(b) TAUAnCMURm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUAnTIS[1:0]	

Table 15-223 TAUAnCMURm settings for the slave channel of the interrupt culling function

Bit name	Setting
TAUAnTIS[1:0]	00: These are not used, so set them to 00.

(c) Channel output mode for the slave channel

Because the channel output mode is not used by this function, clear TAUAnTOE.TAUAnTOEm. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-224 Simultaneous rewrite settings for the slave channel of the interrupt culling function

Bit name	Setting
TAUAnRDE.TAUAnRDEm	1: Enables simultaneous rewrite
TAUAnRDS.TAUAnRDSm	0: The master channel is monitored for the simultaneous rewrite trigger
TAUAnRDM.TAUAnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting 1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
TAUAnRDC.TAUAnRDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.TAUAnRDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for interrupt culling function

Table 15-225 Operating procedure for interrupt culling function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 903*. Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *(5) "Register settings for the slave channel" on page 905*. Set the values of the TAUAnCDRm registers of all channels.	Channel operation is stopped.
Start operation	Set TAUAnTS.TAUAnTSm of the master and slave channels to 1 simultaneously. TAUAnTS.TAUAnTSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUAnIm is generated on the master channel.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.TAUAnRSFm can be read at any time. TAUAnRDT.TAUAnRDTm can be changed during operation.	TAUAnCNTm of the master channel loads TAUAnCDRm, and then counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated. • TAUAnCNTm (master) loads the TAUAnCDRm value, and then continues count operation. • TAUAnCNTm of the slave channel counts down each time INTTAUAnIm is detected on the master channel. When TAUAnCNTm of the slave reaches 0000_H: • INTTAUAnIm (slave) is generated.
Stop operation	Set TAUAnTT.TAUAnTTm of the master and slave channels to 1 simultaneously. TAUAnTT.TAUAnTTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TAUAnTEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(7) Specific timing diagrams

(a) Number of interrupts (master) = number of interrupts (slave)

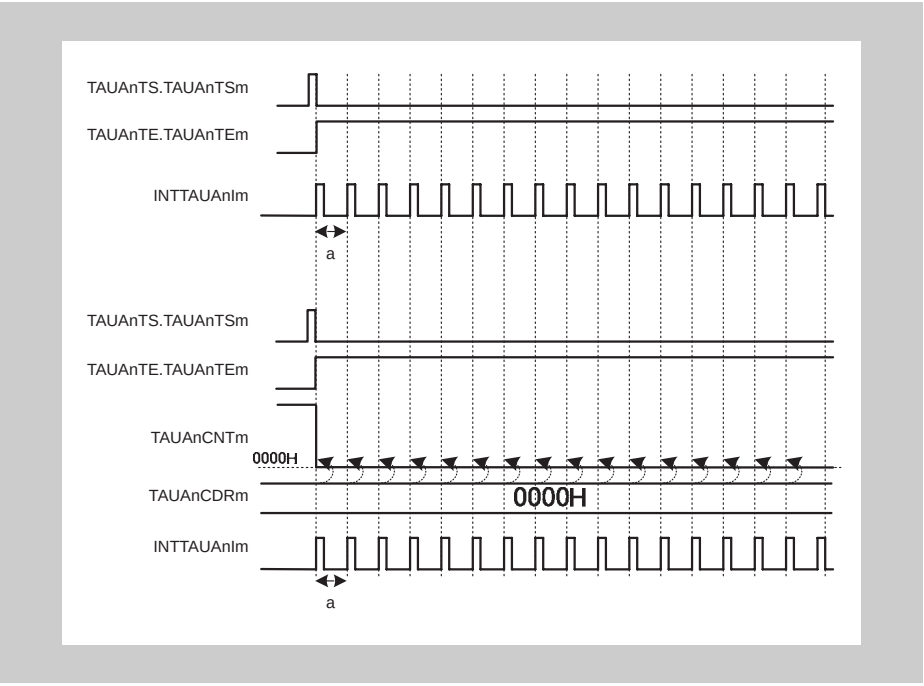


Figure 15-127 TAUAnCDRm (slave) = 0000_H

- If TAUAnCDRm is 0000_H, the TAUAnCDRm value of the slave channel is loaded to TAUAnCNTm each time INTTAUAnIm is detected on the master channel. In other words, TAUAnCNTm is always read as 0000_H.
- Therefore, the slave channel generates an interrupt at the same time as an interrupt is generated on the master channel.

15.28 Registers

This section describes all TAUA registers.

15.28.1 TAUAn register overview

The TAUAn is controlled and operated by the registers in the following table. Where there is one register per channel, this is indicated by an “m”, where m runs from 0 to 15.

Table 15-226 TAUAn register overview (1/2)

Register name	Symbol	Address
TAUAn prescaler registers		
TAUAn prescaler clock select register	TAUAnTPS	<TAUAn_base> + 240_H
TAUAn prescaler baud rate setting register	TAUAnBRS	<TAUAn_base> + 244_H
TAUAn control registers		
TAUAn channel data register m	TAUAnCDRm	<TAUAn_base> + m x 4_H
TAUAn channel counter register m	TAUAnCNTm	<TAUAn_base> + 80_H + m x 4_H
TAUAn channel mode OS register m	TAUAnCMORm	<TAUAn_base> + 200_H + m x 4_H
TAUAn channel mode user register m	TAUAnCMURm	<TAUAn_base> + $C0_H$ + m x 4_H
TAUAn channel status register m	TAUAnCSRm	<TAUAn_base> + 140_H + m x 4_H
TAUAn channel status clear trigger register m	TAUAnCSCm	<TAUAn_base> + 180_H + m x 4_H
TAUAn channel start trigger register	TAUAnTS	<TAUAn_base> + $1C4_H$
TAUAn channel enable status register	TAUAnTE	<TAUAn_base> + $1C0_H$
TAUAn channel stop trigger register	TAUAnTT	<TAUAn_base> + $1C8_H$
TAUAn output registers		
TAUAn channel output enable register	TAUAnTOE	<TAUAn_base> + $5C_H$
TAUAn channel output register	TAUAnTO	<TAUAn_base> + 58_H
TAUAn channel output mode register	TAUAnTOM	<TAUAn_base> + 248_H
TAUAn channel output configuration register	TAUAnTOC	<TAUAn_base> + $24C_H$
TAUAn channel output active level register	TAUAnTOL	<TAUAn_base> + 040_H
TAUAn channel dead time output enable register	TAUAnTDE	<TAUAn_base> + 250_H
TAUAn channel dead time output mode register	TAUAnTDM	<TAUAn_base> + 254_H
TAUAn channel dead time output level register	TAUAnTDL	<TAUAn_base> + 54_H
TAUAn channel real-time output register	TAUAnTRO	<TAUAn_base> + $4C_H$
TAUAn channel real-time output enable register	TAUAnTRE	<TAUAn_base> + 258_H
TAUAn channel real-time output control register	TAUAnTRC	<TAUAn_base> + $25C_H$
TAUAn channel modulation output enable register	TAUAnTME	<TAUAn_base> + 50_H
TAUAn reload data registers		
TAUAn channel reload data enable register	TAUAnRDE	<TAUAn_base> + 260_H
TAUAn channel reload data mode register	TAUAnRDM	<TAUAn_base> + 264_H
TAUAn channel reload data control CH select register	TAUAnRDS	<TAUAn_base> + 268_H
TAUAn channel reload data control register	TAUAnRDC	<TAUAn_base> + $26C_H$
TAUAn channel reload data trigger register	TAUAnRDT	<TAUAn_base> + 44_H

Table 15-226 TAUAn register overview (2/2)

Register name	Symbol	Address
TAUAn channel reload status register	TAUAnRSF	<TAUAn_base> + 48_H
TAUAn DMA window registers		
TAUAn DMA window address setting register 0	TAUAnDAS0	<TAUAn_base> + 270_H
TAUAn DMA window address setting register 1	TAUAnDAS1	<TAUAn_base> + 274_H
TAUAn DMA window address setting register 2	TAUAnDAS2	<TAUAn_base> + 278_H
TAUAn DMA window address setting register 3	TAUAnDAS3	<TAUAn_base> + $27C_H$
TAUAn DMA window address setting register 4	TAUAnDAS4	<TAUAn_base> + 280_H
TAUAn DMA window address setting register 5	TAUAnDAS5	<TAUAn_base> + 284_H
TAUAn DMA window address setting register 6	TAUAnDAS6	<TAUAn_base> + 288_H
TAUAn DMA window address setting register 7	TAUAnDAS7	<TAUAn_base> + $28C_H$
TAUAn DMA window register 0	TAUAnDWR0	<TAUAn_base> + 100_H
TAUAn DMA window register 1	TAUAnDWR1	<TAUAn_base> + 104_H
TAUAn DMA window register 2	TAUAnDWR2	<TAUAn_base> + 108_H
TAUAn DMA window register 3	TAUAnDWR3	<TAUAn_base> + $10C_H$
TAUAn DMA window register 4	TAUAnDWR4	<TAUAn_base> + 110_H
TAUAn DMA window register 5	TAUAnDWR5	<TAUAn_base> + 114_H
TAUAn DMA window register 6	TAUAnDWR6	<TAUAn_base> + 118_H
TAUAn DMA window register 7	TAUAnDWR7	<TAUAn_base> + $11C_H$
TAUAn DMA window register 8	TAUAnDWR8	<TAUAn_base> + 120_H
TAUAn DMA window register 9	TAUAnDWR9	<TAUAn_base> + 124_H
TAUAn DMA window register 10	TAUAnDWR10	<TAUAn_base> + 128_H
TAUAn DMA window register 11	TAUAnDWR11	<TAUAn_base> + $12C_H$
TAUAn DMA window register 12	TAUAnDWR12	<TAUAn_base> + 130_H
TAUAn DMA window register 13	TAUAnDWR13	<TAUAn_base> + 134_H
TAUAn DMA window register 14	TAUAnDWR14	<TAUAn_base> + 138_H
TAUAn DMA window register 15	TAUAnDWR15	<TAUAn_base> + $13C_H$

Note The <TAUAn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

15.28.2 TAUAn prescaler register details

(1) TAUAnTPS - TAUAn prescaler clock select register

This register specifies the PCLK prescalers for clocks CK0, CK1, CK2, and CK3_PRE for all channels. CK3 is generated by dividing CK3_PRE by the factor specified in TAUAnBRS.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 240_H

Initial Value $FFFF_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAnPRS3[3:0]				TAUAnPRS2[3:0]				TAUAnPRS1[3:0]				TAUAnPRS0[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-227 TAUAnTPS register contents (1/4)

Bit position	Bit name	Function
15 to 12	TAUAn PRS3[3:0]	Specifies the CK3_PRE clock. Clock CK3_PRE is the input clock of the BRG unit. The BRG unit supplies the CK3 operation clock for all channels. **PRS3[3:0] / CK3_PRE clock** 0000_B PCLK/2^0 0001_B PCLK/2^1 0010_B PCLK/2^2 0011_B PCLK/2^3 0100_B PCLK/2^4 0101_B PCLK/2^5 0110_B PCLK/2^6 0111_B PCLK/2^7 1000_B PCLK/2^8 1001_B PCLK/2^9 1010_B PCLK/2^{10} 1011_B PCLK/2^{11} 1100_B PCLK/2^{12} 1101_B PCLK/2^{13} 1110_B PCLK/2^{14} 1111_B PCLK/2^{15} These bits can only be rewritten when all counters using CK3 are stopped (TAUAnTE.TAUAnTEm = 0).

Table 15-227 TAUAnTPS register contents (2/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>11 to 8</td><td>TAUAn
PRS2[3:0]</td><td>Specifies the CK2 clock.
<table>
<tr><th>PRS2[3:0]</th><th>CK2 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>0100_B</td><td>PCLK/2^4</td></tr>
<tr><td>0101_B</td><td>PCLK/2^5</td></tr>
<tr><td>0110_B</td><td>PCLK/2^6</td></tr>
<tr><td>0111_B</td><td>PCLK/2^7</td></tr>
<tr><td>1000_B</td><td>PCLK/2^8</td></tr>
<tr><td>1001_B</td><td>PCLK/2^9</td></tr>
<tr><td>1010_B</td><td>PCLK/2^{10}</td></tr>
<tr><td>1011_B</td><td>PCLK/2^{11}</td></tr>
<tr><td>1100_B</td><td>PCLK/2^{12}</td></tr>
<tr><td>1101_B</td><td>PCLK/2^{13}</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK2 are stopped (TAUAnTE.TAUAnTEm = 0).</td></tr>
</table>

Table 15-227 TAUAnTPS register contents (3/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7 to 4</td><td>TAUAn
PRS1[3:0]</td><td>Specifies the CK1 clock.
<table>
<tr><th>PRS1[3:0]</th><th>CK1 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>0100_B</td><td>PCLK/2^4</td></tr>
<tr><td>0101_B</td><td>PCLK/2^5</td></tr>
<tr><td>0110_B</td><td>PCLK/2^6</td></tr>
<tr><td>0111_B</td><td>PCLK/2^7</td></tr>
<tr><td>1000_B</td><td>PCLK/2^8</td></tr>
<tr><td>1001_B</td><td>PCLK/2^9</td></tr>
<tr><td>1010_B</td><td>PCLK/2^{10}</td></tr>
<tr><td>1011_B</td><td>PCLK/2^{11}</td></tr>
<tr><td>1100_B</td><td>PCLK/2^{12}</td></tr>
<tr><td>1101_B</td><td>PCLK/2^{13}</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK1 are stopped (TAUAnTE.TAUAnTEm = 0).</td></tr>
</table>

Table 15-227 TAUAnTPS register contents (4/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>3 to 0</td><td>TAUAn PRS0[3:0]</td><td>Specifies the CK0 clock.
<table>
<tr><th>PRS0[3:0]</th><th>CK0 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>0100_B</td><td>PCLK/2^4</td></tr>
<tr><td>0101_B</td><td>PCLK/2^5</td></tr>
<tr><td>0110_B</td><td>PCLK/2^6</td></tr>
<tr><td>0111_B</td><td>PCLK/2^7</td></tr>
<tr><td>1000_B</td><td>PCLK/2^8</td></tr>
<tr><td>1001_B</td><td>PCLK/2^9</td></tr>
<tr><td>1010_B</td><td>PCLK/2^{10}</td></tr>
<tr><td>1011_B</td><td>PCLK/2^{11}</td></tr>
<tr><td>1100_B</td><td>PCLK/2^{12}</td></tr>
<tr><td>1101_B</td><td>PCLK/2^{13}</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK0 are stopped (TAUAnTE.TAUAnTEm = 0).</td></tr>
</table>

Note The TAUAn clock input PCLK is specified in the first section of this chapter under the keyword “Clock supply”.

(2) TAUAnBRS - TAUAn prescaler baud rate setting register

This register specifies the division factor of prescaler clock CK3.

CK3 is generated by dividing CK3_PRE by the factor specified in this register plus one. The PCLK prescaler for CK3_PRE is specified in TAUAnTPS.TAUAnPRS3[3:0].

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 244_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	TAUAnBRS[07:00]							
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-228 TAUAnBRS register contents

Bit position	Bit name	Function
7 to 0	TAUAn BRS[07:00]	Specifies the CK3_PRE clock division factor for generating CK3:

TAUAnBRS[07:00]	CK3 clock
$0000\ 0000_B$	CK3_PRE / 1
$0000\ 0001_B$	CK3_PRE / 2
$0000\ 0010_B$	CK3_PRE / 3
$0000\ 0011_B$	CK3_PRE / 4
...	...
$1111\ 1110_B$	CK3_PRE / 255
$1111\ 1111_B$	CK3_PRE / 256

15.28.3 TAUAn control register details

(1) TAUAnCDRm - TAUAn channel data register

This register functions either as a compare register or as a capture register, depending on the operation mode specified in TAUAnCMORm.TAUAnMD[4:1].

Access This register can be read/written in 16-bit units.

- In capture mode, only reading is possible. Write operation is ignored.
- In compare mode, reading and writing is possible.

Address <TAUAn_base> + 0_H + m x 4_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAnCDR[15:0]															
R/W															

Table 15-229 TAUAnCDRm register contents

Bit position	Bit name	Function
15 to 0	TAUAn CDR[15:0]	Data register for the capture/compare value.

(2) TAUAnCNTm - TAUAn channel counter register

This register is the channel m counter register.

Access This register can be read in 16-bit units.

Address <TAUAn_base> + 80_H + m x 4_H

Initial Value 0000_H or $FFFF_H$ The initial value depends on the operation mode, see *Table 15-231 "TAUAnCNTm read values after the counter is re-enabled" on page 917*. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAnCNT[15:0]															
R															

Table 15-230 TAUAnCNTm register contents

Bit position	Bit name	Function
15 to 0	TAUAn CNT[15:0]	16-bit counter value.

The read value depends on the counter, the operation mode change, and the values of the TAUAnTS.TAUAnTSm and TAUAnTT.TAUAnTTm bits.

The *initial* counter read value depends on the operation mode and how the counter was stopped:

- by a reset
- by a counter stop trigger (TAUAnTT.TAUAnTTm = 1)

The following table lists the initial counter read values after the counter has stopped (TAUAnTE.TAUAnTEm = 0) and re-enabled (TAUAnTS.TAUAnTSm = 1).

The table also contains the counter read value one count after the counter is enabled (TAUAnTS.TAUAnTSm = 1) for modes where the counter waits for a start trigger.

Table 15-231 TAUAnCNTm read values after the counter is re-enabled (1/2)

Mode name	Count method (up/down)	TAUAnCNTm value		
		After reset	After stop trigger	After one count
Interval timer mode	Count down	$FFFF_H$	Stop value	-
Judge mode	Count down	$FFFF_H$	Stop value	-
Capture mode	Count up	0000_H	Stop value	-
Event count mode	Count down	$FFFF_H$	Stop value	-
One-count mode	Count down	$FFFF_H$	Stop value	$FFFF_H$
Capture & one-count mode	Count up	0000_H	Stop value	Captured value + 1 (TAUAnCDRm)
Judge & one-count mode	Count down	$FFFF_H$	Stop value	TAUAnCNTm value - 1
Up/down count mode	Count up/down	$FFFF_H$	Stop value	-
Pulse one-count mode	Count down	$FFFF_H$	Stop value	0000_H

Table 15-231 TAUAnCNTm read values after the counter is re-enabled (2/2)

Mode name	Count method (up/down)	TAUAnCNTm value		
		After reset	After stop trigger	After one count
Count capture mode	Count up	0000_H	Stop value	-
Gate count mode	Count down	$FFFF_H$	Stop value	Stop value
Capture & gate count mode	Count up	0000_H	Stop value	Stop value

Note If the operation mode is changed while the counter is stopped, the initial counter value after counter restart is undefined. The operation mode is changed by register TAUAnCMORm.TAUAnMD[4:1].

(3) TAUAnCMORm - TAUAn channel mode OS register

This register controls channel m operation.

Access This register can be read or written in 16-bit units. Writing is only possible while the counter is stopped (TAUAnTE.TAUAnTEm = 0).

Address <TAUAn_base> + 200_H + m x 4_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn CKS[1:0]		TAUAn CCS[1:0]		TAUAn MAS	TAUAnSTS[2:0]			TAUAn COS[1:0]		-	TAUAnMD[4:0]				
R/W		R/W		R/W	R/W			R/W		R	R/W				

Table 15-232 TAUAnCMORm register contents (1/4)

Bit position	Bit name	Function
15, 14	TAUAn CKS[1:0]	Selects the operation clock. The operation clock is used for the TAUAnTTINm input edge detection circuit. It can also be used as the count clock depending on bits TAUAnCMORm.TAUAnCCS[1:0]. **TAUAn CKS1 / TAUAn CKS0 / Selected operation clock** 0 / 0 / CK0 0 / 1 / CK1 1 / 0 / CK2 1 / 1 / CK3
13, 12	TAUAn CCS[1:0]	Selects the count clock for TAUAnCNTm counter: **TAUAn CCS1 / TAUAn CCS0 / Selected count clock** 0 / 0 / Operation clock as specified by TAUAnCMORm.TAUAnCKS[1:0]. 0 / 1 / Valid edge of TAUAnTTINm input signal 1 / 0 / Setting prohibited 1 / 1 / INTTAUAnIm signal of the master channel
11	TAUAnMAS	Specifies the channel as master or slave channel during synchronous channel operation: 0: Slave 1: Master This bit is only valid for even channels (CHm_even). For odd channels (CHm_odd), it is fixed to 0.

Table 15-232 TAUAnCMORm register contents (2/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>10 to 8</td><td>TAUAn STS[2:0]</td><td>Selects the external start trigger:
<table>
<tr><th>TAUAn STS2</th><th>TAUAn STS1</th><th>TAUAn STS0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Software trigger</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Valid edge of the TAUAnTTINm input signal. TAUAnCMURm.TAUAnTIS[1:0] specifies the valid edge.</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Valid edge of the TAUAnTTINm input signal is the start trigger and the reverse edge is the stop (capture) trigger</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>INTTAUAnI of the master channel</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>INTTAUAnI of the upper channel (m-1), regardless of the master setting</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>Dead-time output signal of the TAUAnTTOUTm generation unit</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>Up/down output trigger signal TAUAnTUDSm of the master channel.</td></tr>
</table></td></tr>
</table>

Table 15-232 TAUAnCMORm register contents (3/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7, 6</td><td>TAUAn COS[1:0]</td><td>Specifies when the capture register TAUAnCDRm and the overflow flag TAUAnCSRm.TAUAnOVF of channel m are updated.
These bits are only valid if channel m is in capture mode.
<table>
<tr><th>TAUAn COS1</th><th>TAUAn COS0</th><th>TAUAnCDRm</th><th>TAUAnCSRm.TAUAnOVF</th></tr>
<tr><td>0</td><td>0</td><td rowspan="2">Updated upon detection of a TAUAnTTINm input valid edge.</td><td>Updated (cleared or set) upon detection of a TAUAnTTINm input valid edge:
• If a counter overflow has occurred since the last valid edge detection, TAUAnCSRm.TAUAnOVF is set.
• If no counter overflow has occurred since the last valid edge detection, TAUAnCSR.OVF is cleared.</td></tr>
<tr><td>0</td><td>1</td><td>Set upon counter overflow and cleared by setting TAUAnCSCm.TAUAnCLOV.</td></tr>
<tr><td>1</td><td>0</td><td rowspan="2">Updated upon detection of a TAUAnTTINm input valid edge and upon counter overflow:
• TAUAnTTINm input valid edge: Counter value is written to TAUAnCDRm
• Overflow: $FFFF_H$ is written to TAUAnCDRm. The next TAUAnTTINm input valid edge detection is ignored.</td><td>Not set.</td></tr>
<tr><td>1</td><td>1</td><td>Set upon counter overflow and cleared by setting TAUAnCSCm.TAUAnCLOV.</td></tr>
</table></td></tr>
</table>

Table 15-232 TAUAnCMORm register contents (4/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>4 to 0</td><td>TAUAn MD[4:0]</td><td>Specifies the operation mode.
<table>
<tr><th>TAUAn MD4</th><th>TAUAn MD3</th><th>TAUAn MD2</th><th>TAUAn MD1</th><th>TAUAn MD0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1/0</td><td>Interval timer mode</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1/0</td><td>Judge mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1/0</td><td>Capture mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>Event count mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1/0</td><td>One-count mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>1/0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>Capture & one-count mode</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>1/0</td><td>Judge & one-count mode</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>Up/down count mode</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1/0</td><td>Pulse one-count mode</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1/0</td><td>Count capture mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>Gate count mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>Capture & gate count mode</td></tr>
</table>

Mode	Role of the TAUAnMD0 bit
Interval timer mode Capture mode Count capture mode	Specifies whether the INTTAUAnIm signal is output when the counter starts counting (when the start trigger is input). 0: No INTTAUAnIm generated 1: INTTAUAnIm generated
Event count mode Up/down count mode	This bit must be set to 0.
One-count mode Gate count mode Pulse one-count mode	Enables/disables start trigger detection during counting. 0: Disabled 1: Enabled
Capture & one count mode Capture & gate count mode	This bit must be set to 0.
Judge mode Judge one count mode	Specifies when INTTAUAnIm is generated. 0: When TAUAnCNTm ≤ TAUAnCDRm 1: When TAUAnCNTm > TAUAnCDRm
</td></tr>
</table>

(4) TAUAnCMURm - TAUAn channel mode user register

This register specifies the type of valid edge detection used for the TAUAnTTINm input.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + $C0_H$ + m x 4_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	TAUAnTIS[1:0]	
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W

Table 15-233 TAUAnCMURm register contents

Bit position	Bit name	Function
1, 0	TAUAn TIS[1:0]	Specifies the valid edge of the TAUAnTTINm input: • Edge detection for TAUAnTTINm input signals is performed based on the operation clock selected by TAUAnCMORm.TAUAnCKS[1:0].

TAUAn TIS1	TAUAn TIS0	Description
0	0	Falling edge
0	1	Rising edge
1	0	Rising and falling edges (low-width measurement selection). Start trigger: falling edge Stop trigger (capture): rising edge
1	1	Rising and falling edges (high-width measurement selection). Start trigger: rising edge Stop trigger (capture): falling edge

(5) TAUAnCSRm - TAUAn channel status register

This register indicates the count direction and the overflow status of channel m's counter.

Access This register can be read in 16-bit units.

Address <TAUAn_base> + 140_H + m x 4_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	TAUAn CSF	TAUAn OVF
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 15-234 TAUAnCSRm register contents

Bit position	Bit name	Function
1	TAUAn CSF	Indicates the count direction: 0: Counts up 1: Counts down The read value of this bit is only valid in the following mode: • Up/down count mode
0	TAUAn OVF	Indicates the counter overflow status: 0: No overflow occurred 1: Overflow occurred This bit is only used in the following modes: • Capture mode • Capture & one count mode • Count capture mode • Capture & gate count mode The function of this bit depends on the setting of control bits TAUAnCMORm.TAUAnCOS[1:0].

(6) TAUAnCSCm - TAUAn channel status clear register

This register is a trigger register for clearing the overflow flag TAUAnCSRm.TAUAnOVF of a channel m.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUAn_base> + 180_H + m x 4_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	TAUAnCLOV
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Table 15-235 TAUAnCSCm register contents

Bit position	Bit name	Function
0	TAUAnCLOV	0: No function 1: Clears the overflow flag TAUAnCSRm.TAUAnOVF

(7) TAUAnTS - TAUAn channel start trigger register

This register enables the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUAn_base> + $1C4_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAnTS15	TAUAnTS14	TAUAnTS13	TAUAnTS12	TAUAnTS11	TAUAnTS10	TAUAnTS09	TAUAnTS08	TAUAnTS07	TAUAnTS06	TAUAnTS05	TAUAnTS04	TAUAnT03	TAUAnTS02	TAUAnTS01	TAUAnTS00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 15-236 TAUAnTS register contents

Bit position	Bit name	Function
15 to 0	TAUAnTSm	Enables the counter for channel m: 0: No function 1: Enables the counter and sets TAUAnTE.TAUAnTEm = 1. TAUAnTE.TAUAnTEm = 1 only *enables* counter. Whether the counter *starts* depends on the selected operation mode.

(8) TAUAnTE - TAUAn channel enable status register

This register indicates whether counter is enabled or disabled.

Access This register can be read in 16-bit units.

Address <TAUAn_base> + $1C0_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TE15	TAUAn TE14	TAUAn TE13	TAUAn TE12	TAUAn TE11	TAUAn TE10	TAUAn TE09	TAUAn TE08	TAUAn TE07	TAUAn TE06	TAUAn TE05	TAUAn TE04	TAUAn TE03	TAUAn TE02	TAUAn TE01	TAUAn TE00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 15-237 TAUAnTE register contents

Bit position	Bit name	Function
15 to 0	TAUAnTEm	Indicates whether counter for channel m is enabled or disabled: 0: Counter disabled 1: Counter enabled This bit is set when TAUAnTSSTm (the synchronous channel start trigger signal) trigger input is detected or when TAUAnTS.TAUAnTSm is set. Setting TAUAnTT.TAUAnTTm to 1 resets this bit to 0.

(9) TAUAnTT - TAUAn channel stop trigger register

This register stops the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUAn_base> + $1C8_H$

Initial Value 0000_H.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TT15	TAUAn TT14	TAUAn TT13	TAUAn TT12	TAUAn TT11	TAUAn TT10	TAUAn TT09	TAUAn TT08	TAUAn TT07	TAUAn TT06	TAUAn TT05	TAUAn TT04	TAUAn TT03	TAUAn TT02	TAUAn TT01	TAUAn TT00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 15-238 TAUAnTT register contents

Bit position	Bit name	Function
15 to 0	TAUAnTTm	Stops the counter of channel m: 0: No function 1: Stops the counter and reset TAUAnTE.TAUAnTEm. TAUAnCNTm, TAUAnTO.TAUAnTOm, and TAUAnTTOUTm retain their values from before the counter stopped.

15.28.4 TAUAn output register details

(1) TAUAnTOE - TAUAn channel output enable register

This register enables and disables independent channel output mode controlled by software.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + $5C_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TOE15	TAUAn TOE14	TAUAn TOE13	TAUAn TOE12	TAUAn TOE11	TAUAn TOE10	TAUAn TOE09	TAUAn TOE08	TAUAn TOE07	TAUAn TOE06	TAUAn TOE05	TAUAn TOE04	TAUAn TOE03	TAUAn TOE02	TAUAn TOE01	TAUAn TOE00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-239 TAUAnTOE register contents

Bit position	Bit name	Function
15 to 0	TAUAnTOEm	Enables/disables independent channel output mode controlled by software: 0: Enables independent channel output mode controlled by software 1: Disables independent channel output mode controlled by software

(2) TAUAnTOM - TAUAn channel output mode register

This register specifies the output mode of each channel.

Access This register can be read/written in 16-bit units. Writing is only possible while the counter is stopped (TAUAnTE.TAUAnTEm = 0).

Address <TAUAn_base> + 248_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TOM15	TAUAn TOM14	TAUAn TOM13	TAUAn TOM12	TAUAn TOM11	TAUAn TOM10	TAUAn TOM09	TAUAn TOM08	TAUAn TOM07	TAUAn TOM06	TAUAn TOM05	TAUAn TOM04	TAUAn TOM03	TAUAn TOM02	TAUAn TOM01	TAUAn TOM00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-240 TAUAnTOM register contents

Bit position	Bit name	Function
15 to 0	TAUAnTOMm	Specifies the channel output mode: 0: Independent channel output mode 1: Synchronous channel output mode The output mode depends on several channel output control bits, as can be seen in *Table 15-12 “Channel output modes” on page 628*.

(3) TAUAnTOC - TAUAn channel output configuration register

This register specifies the output mode of each channel in combination with TAUAnTOMm.

Access This register can be read/written in 16-bit units. Writing is only possible while the counter is stopped (TAUAnTE.TAUAnTEm = 0).

Address <TAUAn_base> + $24C_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TOC15	TAUAn TOC14	TAUAn TOC13	TAUAn TOC12	TAUAn TOC11	TAUAn TOC10	TAUAn TOC09	TAUAn TOC08	TAUAn TOC07	TAUAn TOC06	TAUAn TOC05	TAUAn TOC04	TAUAn TOC03	TAUAn TOC02	TAUAn TOC01	TAUAn TOC00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-241 TAUAnTOC register contents

Bit position	Bit name	Function
15 to 0	TAUAn TOCm	Specifies the output mode: 0: Operation mode 1 1: Operation mode 2 The output mode also depends on TAUAnTOM.TAUAnTOMm, as can be seen in the following table.

TOMm	TOCm	Description
0	0	Toggle mode: TAUAnTTOUTm toggles when INTTAUAnIm occurs.
	1	Set/reset mode: TAUAnTTOUTm set when INTTAUAnIm occurs upon count start and reset when INTTAUAnIm occurs due to detection of a match between TAUAnCNTm and TAUAnCDRm.
1	0	Synchronous channel operation mode 1: TAUAnTTOUTm set when INTTAUAnI occurs on the master channel and reset when INTTAUAnI occurs on the slave channel.
	1	Synchronous channel operation mode 2: TAUAnTTOUTm set when INTTAUAnIm occurs while the slave channel is counting down and reset when INTTAUAnIm occurs while the slave channel is counting up.

(4) TAUAnTDE - TAUAn channel dead time output enable register

This register enables/disables dead time operation for each channel.

Access This register can be read/written in 16-bit units. Writing is only possible while the counter is stopped (TAUAnTE.TAUAnTEm = 0).

Address <TAUAn_base> + 250_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TDE15	TAUAn TDE14	TAUAn TDE13	TAUAn TDE12	TAUAn TDE11	TAUAn TDE10	TAUAn TDE09	TAUAn TDE08	TAUAn TDE07	TAUAn TDE06	TAUAn TDE05	TAUAn TDE04	TAUAn TDE03	TAUAn TDE02	TAUAn TDE01	TAUAn TDE00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-242 TAUAnTDE register contents

Bit position	Bit name	Function
15 to 0	TAUAnTDEm	Enables/disables dead time control operation of channel m: 0: Disables dead time operation 1: Enables dead time operation The same settings must be set for the even and the odd slave channel that comprise a set. These bits only apply when: • TAUAnTOE.TAUAnTOEm, TAUAnTOM.TAUAnTOMm, and TAUAnTOC.TAUAnTOCm = 1.

(5) TAUAnTDM - TAUAn channel dead time output mode register

This register specifies when dead time is added during dead time output.

Access This register can be read/written in 16-bit units. Writing is only possible while the counter is stopped (TAUAnTE.TAUAnTEm = 0).

Address <TAUAn_base> + 254_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TDM15	TAUAn TDM14	TAUAn TDM13	TAUAn TDM12	TAUAn TDM11	TAUAn TDM10	TAUAn TDM09	TAUAn TDM08	TAUAn TDM07	TAUAn TDM06	TAUAn TDM05	TAUAn TDM04	TAUAn TDM03	TAUAn TDM02	TAUAn TDM01	TAUAn TDM00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-243 TAUAnTDM register contents

Bit position	Bit name	Function
15 to 0	TAUAnTDMm	Specifies when dead time is added during dead time output: 0: Upon detection of a duty cycle at the upper even channel (duty dead time output) 1: Upon detection of a TAUAnTTIN input edge at the lower odd channel (one-phase dead time output) The same settings must be set for the even and the odd slave channel that comprise a set. These bits only apply when: • TAUAnTOE.TAUAnTOEm, TAUAnTOM.TAUAnTOMm, TAUAnTOC.TAUAnTOCm, and TAUAnTDE.TAUAnTDEm = 1.

(6) TAUAnTDL - TAUAn channel dead time output level register

This register selects the phase period to which dead time is added.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 54_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TDL15	TAUAn TDL14	TAUAn TDL13	TAUAn TDL12	TAUAn TDL11	TAUAn TDL10	TAUAn TDL09	TAUAn TDL08	TAUAn TDL07	TAUAn TDL06	TAUAn TDL05	TAUAn TDL04	TAUAn TDL03	TAUAn TDL02	TAUAn TDL01	TAUAn TDL00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-244 TAUAnTDL register contents

Bit position	Bit name	Function
15 to 0	TAUAnTDLm	Selects the phase period to which dead time is added: 0: Positive phase period 1: Negative phase period These bits only apply when: • TAUAnTOE.TAUAnTOEm, TAUAnTOM.TAUAnTOMm, TAUAnTOC.TAUAnTOCm, and TAUAnTDE.TAUAnTDEm = 1.

(7) TAUAnTRE - TAUAn channel real-time output enable register

This register enables or disables real-time output.

Access This register can be read/written in 16-bit units. Writing is only possible while TAUAnTE.TAUAnTEm is 0.

Address <TAUAn_base> + 258_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TRE15	TAUAn TRE14	TAUAn TRE13	TAUAn TRE12	TAUAn TRE11	TAUAn TRE10	TAUAn TRE09	TAUAn TRE08	TAUAn TRE07	TAUAn TRE06	TAUAn TRE05	TAUAn TRE04	TAUAn TRE03	TAUAn TRE02	TAUAn TRE01	TAUAn TRE00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-245 TAUAnTRE register contents

Bit position	Bit name	Function
15 to 0	TAUAnTREm	Enables or disables real-time output of channel m. 0: Disables real-time output 1: Enables real-time output These bits only apply when TAUTnTOE.TAUAnTOEm = 1. When TAUAnTRE.TAUAnTREm = 0, TAUAnTTOUTm is not affected by real-time output. When TAUAnTRE.TAUAnTREm = 1, TAUAnTTOUTm outputs the value of the real-time output bit TAUAnTRO.TAUAnTROm, dependent on the timer operation.

(8) TAUAnTRC - TAUAn channel real-time control register

This register controls the real-time output trigger for each channel.

Access This register can be read/written in 16-bit units. Writing is only possible while TAUAnTE.TAUAnTEm is 0.

Address <TAUAn_base> + $1C0_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TRC15	TAUAn TRC14	TAUAn TRC13	TAUAn TRC12	TAUAn TRC11	TAUAn TRC10	TAUAn TRC09	TAUAn TRC08	TAUAn TRC07	TAUAn TRC06	TAUAn TRC05	TAUAn TRC04	TAUAn TRC03	TAUAn TRC02	TAUAn TRC01	TAUAn TRC00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-246 TAUAnTRC register contents

Bit position	Bit name	Function
15 to 0	TAUAnTRCm	Specifies which channel generates the real-time output trigger for channel m: 0: The next upper channel where this bit is set to 1 1: Channel m These bits only apply when TAUAnTRE.TAUAnTREm = 1.

(9) TAUAnTRO - TAUAn channel real-time output register

For this register, specify the value output to TAUAnTTOUTm.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + $04C_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TRO15	TAUAn TRO14	TAUAn TRO13	TAUAn TRO12	TAUAn TRO11	TAUAn TRO10	TAUAn TRO09	TAUAn TRO08	TAUAn TRO07	TAUAn TRO06	TAUAn TRO05	TAUAn TRO04	TAUAn TRO03	TAUAn TRO02	TAUAn TRO01	TAUAn TRO00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-247 TAUAnTRO register contents

Bit position	Bit name	Function
15 to 0	TAUAnTROm	Specifies the value output to TAUAnTTOUTm. When TAUAnTRE.TAUAnTRE is 0, the TAUAnTROm value is not output to TAUAnTTOUTm even if a real-time trigger is generated.

(10) TAUAnTME - TAUAn channel modulation output enable register

This register enables and disables modulation output for the timer output and real-time output.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 050_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TME15	TAUAn TME14	TAUAn TME13	TAUAn TME12	TAUAn TME11	TAUAn TME10	TAUAn TME09	TAUAn TME08	TAUAn TME07	TAUAn TME06	TAUAn TME05	TAUAn TME04	TAUAn TME03	TAUAn TME02	TAUAn TME01	TAUAn TME00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-248 TAUAnTME register contents

Bit position	Bit name	Function
15 to 0	TAUAnTMEm	Enables/disables modulation output for timer output and real-time output of channel m: 0: Disables modulation 1: Enables modulation These bits only apply when TAUAnTOE.TAUAnTOEm and TAUAnTRE.TAUAnTREm = 1.

15.28.5 TAUAn channel output level register details

(1) TAUAnTO - TAUAn channel output register

This register specifies and reads the level of TAUAnTTOUTm.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 58_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TO15	TAUAn TO14	TAUAn TO13	TAUAn TO12	TAUAn TO11	TAUAn TO10	TAUAn TO09	TAUAn TO08	TAUAn TO07	TAUAn TO06	TAUAn TO05	TAUAn TO04	TAUAn TO03	TAUAn TO02	TAUAn TO01	TAUAn TO00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-249 TAUAnTO register contents

Bit position	Bit name	Function
15 to 0	TAUAnTOm	Specifies/reads the level of TAUAnTTOUTm: 0: Low 1: High Only TAUAnTOm bits for which independent channel output function is disabled (TAUAnTOEm = 0) can be written.

(2) TAUAnTOL - TAUAn channel output level register

This register specifies the output logic of the channel output bit (TAUAnTO.TAUAnTOm).

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 040_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn TOL15	TAUAn TOL14	TAUAn TOL13	TAUAn TOL12	TAUAn TOL11	TAUAn TOL10	TAUAn TOL09	TAUAn TOL08	TAUAn TOL07	TAUAn TOL06	TAUAn TOL05	TAUAn TOL04	TAUAn TOL03	TAUAn TOL02	TAUAn TOL01	TAUAn TOL00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-250 TAUAnTOL register contents

Bit position	Bit name	Function
15 to 0	TAUAnTOLm	Specifies the output logic of the channel m output bit (TAUAnTO.TAUAnTOm): 0: Positive logic (active high) 1: Inverted logic (active low)

15.28.6 TAUAn simultaneous rewrite register details

(1) TAUAnRDE - TAUAn channel reload data enable register

This register enables and disables simultaneous rewriting of the data register TAUAnCDRm or TAUAnTOLm.

Access This register can be read/written in 16-bit units. Writing is only possible while TAUAnTE.TAUAnTEm is 0.

Address <TAUAn_base> + $26C_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn RDE15	TAUAn RDE14	TAUAn RDE13	TAUAn RDE12	TAUAn RDE11	TAUAn RDE10	TAUAn RDE09	TAUAn RDE08	TAUAn RDE07	TAUAn RDE06	TAUAn RDE05	TAUAn RDE04	TAUAn RDE03	TAUAn RDE02	TAUAn RDE01	TAUAn RDE00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-251 TAUAnRDE register contents

Bit position	Bit name	Function
15 to 0	TAUAnRDEm	Enables/disables simultaneous rewrite of the data register of channel m: 0: Disables simultaneous rewrite 1: Enabled simultaneous rewrite

(2) TAUAnRDS - TAUAn channel reload data control channel select register

This register selects the control channel for simultaneous rewrite.

Access This register can be read/written in 16-bit or 1-bit units. Writing is only possible while TAUAnTE.TAUAnTEm is 0.

Address <TAUAn_base> + 268_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn RDS15	TAUAn RDS14	TAUAn RDS13	TAUAn RDS12	TAUAn RDS11	TAUAn RDS10	TAUAn RDS09	TAUAn RDS08	TAUAn RDS07	TAUAn RDS06	TAUAn RDS05	TAUAn RDS04	TAUAn RDS03	TAUAn RDS02	TAUAn RDS01	TAUAn RDS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-252 TAUAnRDS register contents

Bit position	Bit name	Function
15 to 0	TAUAnRDSm	Specifies which channel is monitored for the simultaneous rewrite trigger: 0: Master channel 1: Another upper channel

(3) TAUAnRDM - TAUAn channel reload data mode register

This register selects when the signal that controls simultaneous rewrite is loaded.

Access This register can be read/written in 16-bit units. Writing is only possible while TAUAnTE.TAUAnTEm is 0.

Address <TAUAn_base> + 264_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn RDM15	TAUAn RDM14	TAUAn RDM13	TAUAn RDM12	TAUAn RDM11	TAUAn RDM10	TAUAn RDM09	TAUAn RDM08	TAUAn RDM07	TAUAn RDM06	TAUAn RDM05	TAUAn RDM04	TAUAn RDM03	TAUAn RDM02	TAUAn RDM01	TAUAn RDM00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-253 TAUAnRDM register contents

Bit position	Bit name	Function
15 to 0	TAUAnRDMm	Selects when the signal that triggers simultaneous is generated: 0: When the master channel counter starts counting 1: At the top of a triangle wave cycle These bits only apply when TAUAnRDE.TAUAnRDEm = 1 and TAUAnRDS.RDSm = 0.

(4) TAUAnRDC - TAUAn channel reload data control register

This register specifies the channel that generates the INTTAUAnIm signal that triggers simultaneous rewrite.

Access This register can be read/written in 16-bit units. Writing is only possible while TAUAnTE.TAUAnTEm is 0.

Address <TAUAn_base> + $26C_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn RDC15	TAUAn RDC14	TAUAn RDC13	TAUAn RDC12	TAUAn RDC11	TAUAn RDC10	TAUAn RDC09	TAUAn RDC08	TAUAn RDC07	TAUAn RDC06	TAUAn RDC05	TAUAn RDC04	TAUAn RDC03	TAUAn RDC02	TAUAn RDC01	TAUAn RDC00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-254 TAUAnRDC register contents

Bit position	Bit name	Function
15 to 0	TAUAnRDCm	Specifies whether the channel generates the simultaneous rewrite trigger signal. 0: Do not have the channel generate the simultaneous rewrite trigger signal. 1: Have the channel generate the simultaneous rewrite trigger signal. These bits only apply when TAUAnRDS.TAUAnRDSm is 1.

(5) TAUAnRDT - TAUAn channel reload data trigger register

This register triggers the simultaneous rewrite pending state.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUAn_base> + 044_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn RDT15	TAUAn RDT14	TAUAn RDT13	TAUAn RDT12	TAUAn RDT11	TAUAn RDT10	TAUAn RDT09	TAUAn RDT08	TAUAn RDT07	TAUAn RDT06	TAUAn RDT05	TAUAn RDT04	TAUAn RDT03	TAUAn RDT02	TAUAn RDT01	TAUAn RDT00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 15-255 TAUAnRDT register contents

Bit position	Bit name	Function
15 to 0	TAUAnRDTm	Triggers the simultaneous rewrite pending state: 0: No function 1: Simultaneous rewrite pending state is triggered. The simultaneous rewrite pending flag (TAUAnRSFm) is set to 1. The system waits for the simultaneous rewrite trigger.

(6) TAUAnRSF - TAUAn channel reload status register

This flag register indicates the simultaneous rewriting status.

Access This register can be read in 16-bit units.

Address <TAUAn_base> + 048_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn RSF15	TAUAn RSF14	TAUAn RSF13	TAUAn RSF12	TAUAn RSF11	TAUAn RSF10	TAUAn RSF09	TAUAn RSF08	TAUAn RSF07	TAUAn RSF06	TAUAn RSF05	TAUAn RSF04	TAUAn RSF03	TAUAn RSF02	TAUAn RSF01	TAUAn RSF00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 15-256 TAUAnRSF register contents

Bit position	Bit name	Function
15 to 0	TAUAnRSFm	Indicates the simultaneous rewrite status: 0: Simultaneous rewriting was performed due to the generation of a simultaneous rewrite trigger. 1: Simultaneous rewriting is pending (TAUAnRDTm = 1).

15.28.7 TAUAn DMA window registers

(1) TAUAnDASi - TAUAn DMA window address setting register i (i = 0 to 7)

This register specifies addresses of the window registers used for DMA. Eight TAUAnDASi registers control sixteen TAUAnDWRm registers, i.e. each TAUAnDASi register controls two TAUAnDWRm registers independently.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 0270_H + m x 4_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUAn DAS *odd7*	TAUAn DAS *odd6*	TAUAn DAS *odd5*	TAUAn DAS *odd4*	TAUAn DAS *odd3*	TAUAn DAS *odd2*	TAUAn DAS *odd1*	TAUAn DAS *odd0*	TAUAn DAS *even7*	TAUAn DAS *even6*	TAUAn DAS *even5*	TAUAn DAS *even4*	TAUAn DAS *even3*	TAUAn DAS *even2*	TAUAn DAS *even1*	TAUAn DAS *even0*
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

odd = 01, 03, 05, 07, 09, 11, 13, 15

even = 00, 02, 04, 06, 08, 10, 12, 14

The following table shows the relationship between the TAUAnDAS control registers and the TAUAnDWRm registers:

Table 15-257 Relationship between TAUAnDAS registers and TAUAnDWRm registers

Control register	Control bits	Control target register TAUAnDWRm
TAUAnDAS0	Bits 7 to 0	TAUAnDWR0
TAUAnDAS0	Bits 15 to 8	TAUAnDWR1
TAUAnDAS1	Bits 7 to 0	TAUAnDWR2
TAUAnDAS1	Bits 15 to 8	TAUAnDWR3
TAUAnDAS2	Bits 7 to 0	TAUAnDWR4
TAUAnDAS2	Bits 15 to 8	TAUAnDWR5
TAUAnDAS3	Bits 7 to 0	TAUAnDWR6
TAUAnDAS3	Bits 15 to 8	TAUAnDWR7
TAUAnDAS4	Bits 7 to 0	TAUAnDWR8
TAUAnDAS4	Bits 15 to 8	TAUAnDWR9
TAUAnDAS5	Bits 7 to 0	TAUAnDWR10
TAUAnDAS5	Bits 15 to 8	TAUAnDWR11
TAUAnDAS6	Bits 7 to 0	TAUAnDWR12
TAUAnDAS6	Bits 15 to 8	TAUAnDWR13
TAUAnDAS7	Bits 7 to 0	TAUAnDWR14
TAUAnDAS7	Bits 15 to 8	TAUAnDWR15

Table 15-258 TAUAnDASm register contents

Bit position	Bit name	Function
15 to 8	TAUAnDAS *odd*[15:8]	00_H to $3C_H$: Specifies the CDR0 to CDR15 registers. 40_H: Specifies the TAUAnTOL register. 44_H: Specifies the TAUAnRDT register. 48_H: Specifies the TAUAnRSF register. $4C_H$: Specifies the TAUAnTRO register. 50_H: Specifies the TAUAnTME register. 54_H: Specifies the TAUAnTDL register. 58_H: Specifies the TAUAnTO register. $5C_H$: Specifies the TAUAnTOE register. 60_H to $7C_H$: Setting prohibited 80_H to BC_H: Specifies the CNT0 to CNT15 registers. $C0_H$ to FC_H: Setting prohibited
7 to 0	TAUAnDAS *even*[7:0]	00_H to $3C_H$: Specifies the CDR0 to CDR15 registers. 40_H: Specifies the TAUAnTOL register. 44_H: Specifies the TAUAnRDT register. 48_H: Specifies the TAUAnRSF register. $4C_H$: Specifies the TAUAnTRO register. 50_H: Specifies the TAUAnTME register. 54_H: Specifies the TAUAnTDL register. 58_H: Specifies the TAUAnTO register. $5C_H$: Specifies the TAUAnTOE register. 60_H to $7C_H$: Setting prohibited 80_H to BC_H: Specifies the CNT0 to CNT15 registers. $C0_H$ to FC_H: Setting prohibited

Be sure to fix bits [9:8] and [1:0] to “0”.

(2) TAUAnDWRm - TAUAn DMA window register m

This register is used for DMA (m = 0 to 15). TAUAnDWRm mirrors the addresses specified for the corresponding TAUAnDASi registers (where i is a value from 0 to 7). (See *(1) “TAUAnDASi - TAUAn DMA window address setting register i (i = 0 to 7)”*.)

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 0100_H + m x 4_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA window address specified for TAUAnDASi (where i is a value from 0 to 7)															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Chapter 16 Timer Array Unit B (TAUB)

This chapter describes timer array unit B (TAUB).

The first section describes all V850E2/Sx4-H specific properties, such as instances, register base addresses, and input/output signal names.
The subsequent sections describe the features that apply to all implementations.

16.1 V850E2/Sx4-H TAUB Features

Instances

This microcontroller has following number of instances of TAUB:

Table 16-1 Instances of TAUB

TAUB	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of instances	0	0	1
Name	–	–	TAUB2

Instances index n

Throughout this chapter, the individual instances of TAUB is identified by the index "n" (n = 2), for example, TAUBnTOM for the TAUBn channel output mode register.

Channel index m

TAUB has 16 channels. Throughout this chapter, the individual channels are identified by the index “m” (m = 0 to 15), thus a certain channel is denoted as CHm.
The even numbered channels (m = 0, 2, 4, 6, 8, 10, 12, 14) are denoted as CHm_even.
The odd numbered channels (m = 1, 3, 5, 7, 9, 11, 13, 15) are denoted as CHm_odd.

Register addresses

All TAUBn register addresses are given as addresses offset fromthe individual base address <TAUBn_base>.
The base address <TAUBn_base> of each TAUBn is listed in the following table:

Table 16-2 Register base addresses <TAUBn_base>

TAUBn	<TAUBn_base> address
TAUB2	FF80 $A000_H$

Clock supply

The following clock is supplied to TAUB:

Table 16-3 TAUBn clock supply

TAUBn	Clock	Connected to:
TAUB2	PCLK	Clock generator CKSCLK_111

Interrupts and DMA

TAUB can generate the following interrupt and DMA requests:

Table 16-4 TAUBn interrupt and DMA requests

TAUBn signals	Function	Connected to:
TAUB2:		
INTTAUB2I0	Channel 0 interrupt	Interrupt generator INTTAUB2I0 DMA controller trigger 20
INTTAUB2I1	Channel 1 interrupt	Interrupt generator INTTAUB2I1 DMA controller trigger 21
INTTAUB2I2	Channel 2 interrupt	Interrupt generator INTTAUB2I2 DMA controller trigger 107
INTTAUB2I3	Channel 3 interrupt	Interrupt generator INTTAUB2I3 DMA controller trigger 108
INTTAUB2I4	Channel 4 interrupt	Interrupt generator INTTAUB2I4 DMA controller trigger 22
INTTAUB2I5	Channel 5 interrupt	Interrupt generator INTTAUB2I5 DMA controller trigger 23
INTTAUB2I6	Channel 6 interrupt	Interrupt generator INTTAUB2I6 DMA controller trigger 109
INTTAUB2I7	Channel 7 interrupt	Interrupt generator INTTAUB2I7 DMA controller trigger 110
INTTAUB2I8	Channel 8 interrupt	Interrupt generator INTTAUB2I8 DMA controller trigger 24
INTTAUB2I9	Channel 9 interrupt	Interrupt generator INTTAUB2I9 DMA controller trigger 25
INTTAUB2I10	Channel 10 interrupt	Interrupt generator INTTAUB2I10 DMA controller trigger 111
INTTAUB2I11	Channel 11 interrupt	Interrupt generator INTTAUB2I11 DMA controller trigger 112
INTTAUB2I12	Channel 12 interrupt	Interrupt generator INTTAUB2I12 DMA controller trigger 26
INTTAUB2I13	Channel 13 interrupt	Interrupt generator INTTAUB2I13 DMA controller trigger 27
INTTAUB2I14	Channel 15 interrupt	Interrupt generator INTTAUB2I14 DMA controller trigger 113
INTTAUB2I15	Channel 15 interrupt	Interrupt generator INTTAUB2I15 DMA controller trigger 114

TAUB hardware reset

TAUB and their registers are initialized by the following reset signal:

Table 16-5 TAUBn reset signal

TAUBn	Reset signal
TAUBn	System reset SYSRES

I/O signals

The I/O signals of TAUB are listed in the following table:

Table 16-6 TAUBn I/O signals

TAUB signal	Function	Connected to:
TAUB2:		
TAUB2TTIN0 to TAUB2TTIN15	Channel 0 to 15 input	Port TAUB2I0 to TAUB2I15
TAUB1TTOUT0 to TAUB2TTOUT15	Channel 0 to 15 output	Port TAUB2O0 to TAUB2O15

All TAUBn interrupt and I/O signals are sketched in the following figure.

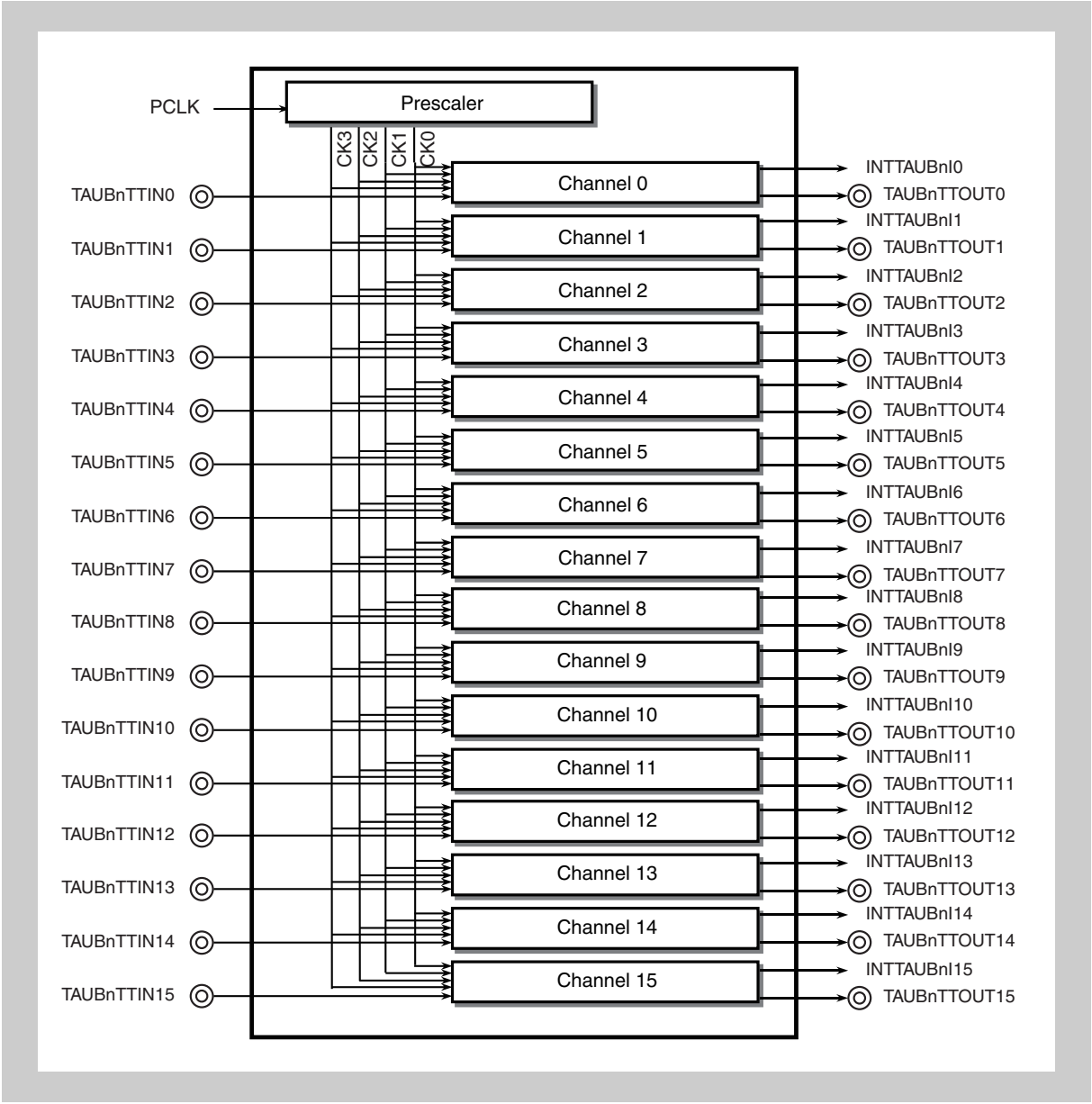


Figure 16-1 TAUB I/O and interrupt signals

16.2 Functional Overview

Features summary

The TAUB has the following functions:

- 16 channels
- 16-bit counter and 16-bit data register per channel
- Independent channel operation
- Synchronous channel operation (master and slave operation)
- Generation of different types of output signal
- Counter can be triggered by external signal
- Interrupt generation

The following figure shows the main components of the TAUB:

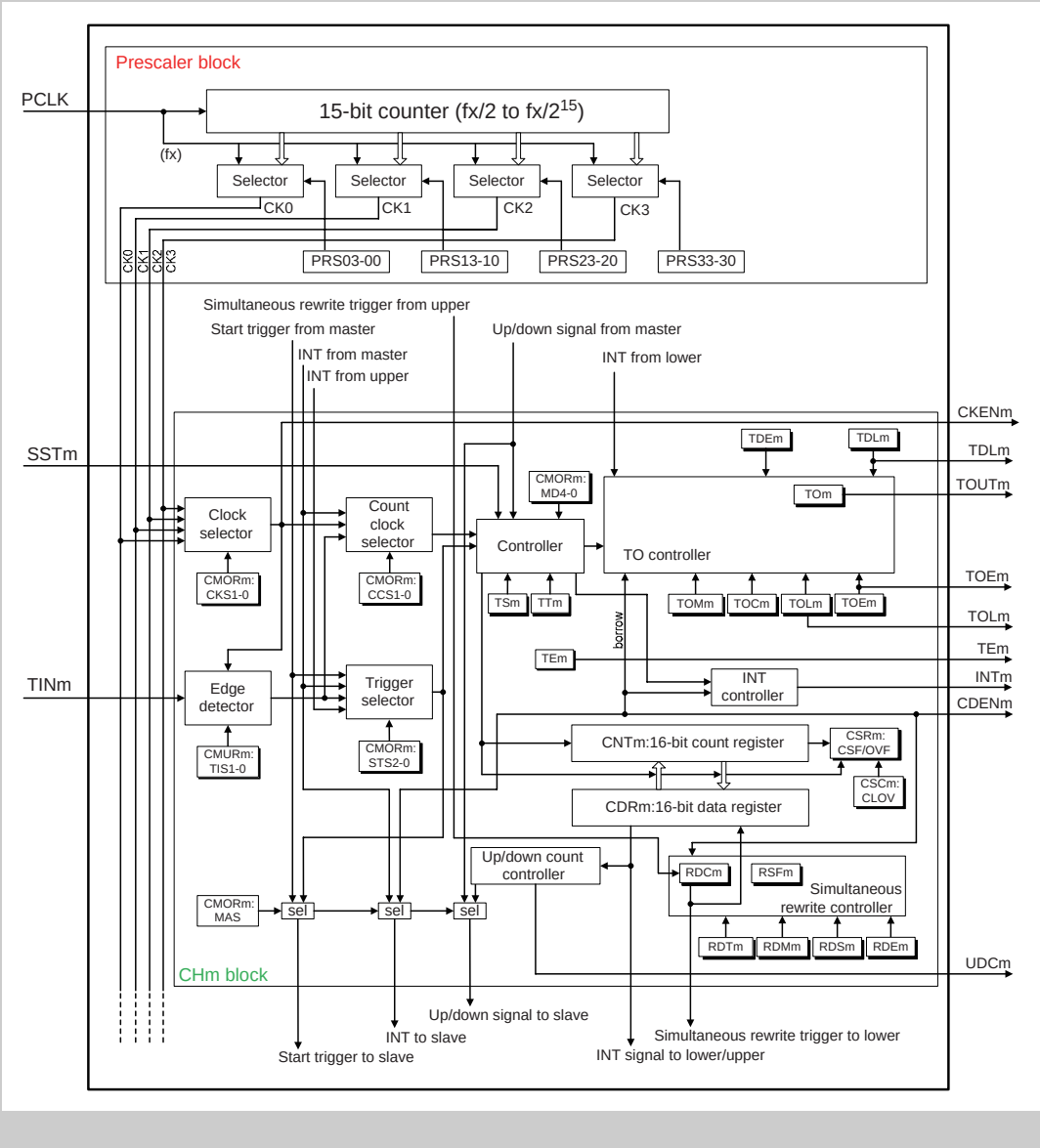


Figure 16-2 Block diagram of the TAUB

The prefix “TAUBn” has been omitted from the register names for the sake of clarity in the above figure.

16.2.1 Terms

In this chapter, the following terms are used:

- **Independent / synchronous channel operation**

  Independent or synchronous channel operation describes the dependency of channels on each other:

  – If a channel operates independent of all other channels, this is called independent channel operation.

  – If a channel operates depending on other channels, this is called synchronous channel operation.

- **Channel group**

  In synchronous channel operation, all channels that depend on each other are referred to as a “channel group”.
  A channel group has one master channel and one or more slave channels.

- **Operation mode**

  An operation mode can be selected for every channel m. The operation mode defines the *basic* operation and features of a channel.
  In synchronous channel operation, every channel in the channel group can operate in a different operation mode.
  Examples are capture mode, event count mode, and interval timer mode.

- **Channel output mode**

  The channel output mode defines the operation of TAUBnTTOUTm

  – of a single channel (independent output operation) or

  – of all channels in a channel group (synchronous output operation).

  Examples are independent channel output mode and synchronous channel output mode with dead time.

- **Channel operation function**

  The channel operation function defines the *complete* function and all features

  – of a single channel (independent channel operation) or

  – of all channels in a channel group (synchronous channel operation).

  It defines the operation mode, start and capture trigger, count clock, the real-time trigger generation, and the channel output mode.
  Examples are the clock divider function and triangle PWM output function.

- **Upper / lower channel**

  Depending on the channel number m, a neighboring channel can be referred to as “upper” or “lower” channel:

  – Upper channel: Channel with a smaller channel number

  – Lower channel: Channel with a higher channel number

  Example:
  For channel 5, channel 3 is an upper channel and channel 9 is a lower channel.

16.3 Functional Description

The Timer Array Unit B is used to perform various count or timer operations and to output a signal which depends on the result of the operation. It contains one prescaler block for count clock generation and 16 channels, each equipped with a 16 bit counter TAUBnCNTm and a 16-bit data register TAUBnCDRm to hold the start or compare value of the counter.

It also contains several control and status registers.

Independent and synchronous operation

Every channel can operate in different operation modes, either independently or in combination with other channels (synchronously), i.e. multiple channels depend on each other with one master and one or more slave channels.

When a channel is operated independently, its operation mode and functions are not affected by those of other channels. When a channel is operated synchronously it is either a master or a slave. A master channel can have multiple slaves, and the state of one channel affects that of the other channels. For example, this means that one channel can control when another starts to count, is reset, etc.

The following describes the functional blocks:

Prescaler block

The prescaler block provides up to 4 clock signals (CK0 to CK3) that can be used as count clocks for all channels.

Count clocks CK0 to CK3 are derived from PCLK by a configurable prescaler division factor of 2^0 to 2^{15}.

Clock and count clock selection

For every channel, the count clock selector selects which of the following is used as the clock source:

- One of the clocks CK0 to CK3 (selected by the clock selector)
- INTTAUBnIm from master channel
- Edge detected TAUBnTTINm input signal

Controller

The controller controls the main operations of the counter:

- Operation mode (selected by bits TAUBnCMORm.MD[4:0])
- Counter start enable (TAUBnTS.TSm) and counter stop (TAUBnTT.TTm)

  When counter start is enabled, status flag TAUBnTE.TEm is set.
- Count direction (can be controlled by master channel)

Trigger selector

Depending on the selected operation mode, the counter starts automatically when it is enabled (TAUBnTE.TEm = 1), or it waits for an external start trigger signal. Any of the following signals can be used as the start trigger:

- Synchronous channel start trigger input TAUBnTSSTm
- TAUBnTTINm input valid edge
- INTTAUBnIm from the master or any upper channel
- Up/down output trigger signal TAUBnTUDSm of the master channel
- Dead-time output signal TAUBnTDL.TDLm of the TAUBnTTOUTm generation unit.

Simultaneous rewrite controller

Simultaneous rewrite control is a special function that can be used in synchronous operation modes. The data registers of all channels in a channel group can be rewritten at any time. The simultaneous rewrite controller

ensures that new data register values of all channels become effective at the same time.

TAUBnTO controller The output control of every channel enables the generation of various output signal forms such as PWM signals or triangular waves.

Signals The TAUB has various input and output signals. A full list can be found in the first section of this chapter under the keyword “I/O signals”.

16.3.1 Timer operation functions

The functions below can be achieved by operating TAUB independently on each channel or by operating it on a combination of multiple channels.

Table 16-7 TAUB operation functions

Independent channel operation function	Synchronous channel operation function
Independent channel operation functions	Synchronous channel operation functions
Interval timer function	PWM output function
TAUBnTTINm input interval timer function	Delay pulse output function
One-pulse output function	A/D conversion trigger output function type 1
Independent channel signal measurement functions	Synchronous PWM signal functions triggered by an external signal
TAUBnTTINm input pulse interval measurement function	One-shot pulse output function
TAUBnTTINm input signal width measurement function	Synchronous triangle PWM output function
Overflow interrupt output function (during TAUBnTTINm width measurement)	Triangle PWM output function
TAUBnTTINm input period count detection function	Triangle PWM output function with dead time
Overflow interrupt output function (during TAUBnTTINm input period count detection)	A/D conversion trigger output function type 2
TAUBnTTINm input pulse interval judgment function	–
TAUBnTTINm input signal width judgment function	–
Independent channel simultaneous rewrite functions	–
Simultaneous rewrite trigger generation function type 1	–
Other independent channel functions	–
External event count function	–
Clock divide function	–
TAUBnTTINm input position detection function	–

16.4 General Operating Procedure

The following lists the general operation procedure for the TAUBn:

After reset release, the operation of each channel is stopped. Clock supply is started and writing to each register is enabled. All circuits and registers of all channels are initialized. The control register of TAUBnTTOUTm is also initialized and outputs a low level.

1. Set the TAUBnTPS register to specify the clock frequency of CK0 to CK3.
2. Configure the desired TAUBn function:
   – Set the operation mode
   – Set the channel output mode
   – Set any other control bits
3. Enable the counter by setting the TAUBnTS.TSm bit to 1.

   The counter starts to count immediately, or when an appropriate trigger is detected, depending on the bit settings.
   The function is in operation.
4. If desired, and if possible for the configured function, stop the counter or perform a forced restart operation.
5. Stop the function by setting the TAUBnTT.TTm bit to 0.

Note A detailed description of the required control bits and the operation of the individual functions is given in *16.13 “Independent Channel Operation Functions” on page 979* and *16.18 “Synchronous Channel Operation Functions” on page 1070*.

16.5 Operation Modes

The TAUB contains 12 operation modes. These determine the basic behavior of a channel, for example whether a timer counts up or down, whether the data register TAUBnCDRm acts as a compare register or stores the initial value of the counter, etc.

One operation mode can be set for each channel. It is specified using the TAUBnCMOR.MD[4:0] bits.

When choosing a function, these settings cannot be specified individually, but are grouped under the term operation mode. If a function uses multiple channels, the operation mode of each channel must be set correctly for the function to work correctly.

For details about the operation modes required by each function, see the required function in *16.13 “Independent Channel Operation Functions” on page 979* and *16.18 “Synchronous Channel Operation Functions” on page 1070*.

16.6 Concepts of Synchronous Channel Operation

In synchronous channel operation, multiple channels depend on each other, or are affected by changes in another channel. Therefore, several rules apply for the use of synchronous channel functions. These rules are detailed in *16.6.1 "Rules"*.

Two special features for synchronous channel operation are detailed in the following subchapters:

- *16.6.2 "Simultaneous start and stop of synchronous channel counters" on page 951*
- *16.7 "Simultaneous Rewrite" on page 952*

16.6.1 Rules

Number of masters and slaves

- Only even channels (CH0, CH2, CH4, …) can be set as master channels. Any channel apart from CH0 can be set as a slave channel.
- Only channels lower than the master channel can be set as slave channels, and several slave channels can be set for one master channel.
  Example: If CH2 is a master channel, CH3 and the lower channels (CH4, CH5, …) can be set as slave channels.
- If multiple master channels are used, slave channels cannot cross the master channels.
  Example: If CH0 and CH4 are master channels, CH1 to CH3 can be set as slave channels for CH0, but CH5 to CH7 cannot.

Operation clock

- The same operation clock must be set for the slave channel and the master channel. This is achieved using the TAUBnCMORm.CKS[1:0] bits of the slave and master channel.

The basic concepts of master/slave usage and operation clocks are illustrated in the following figure.

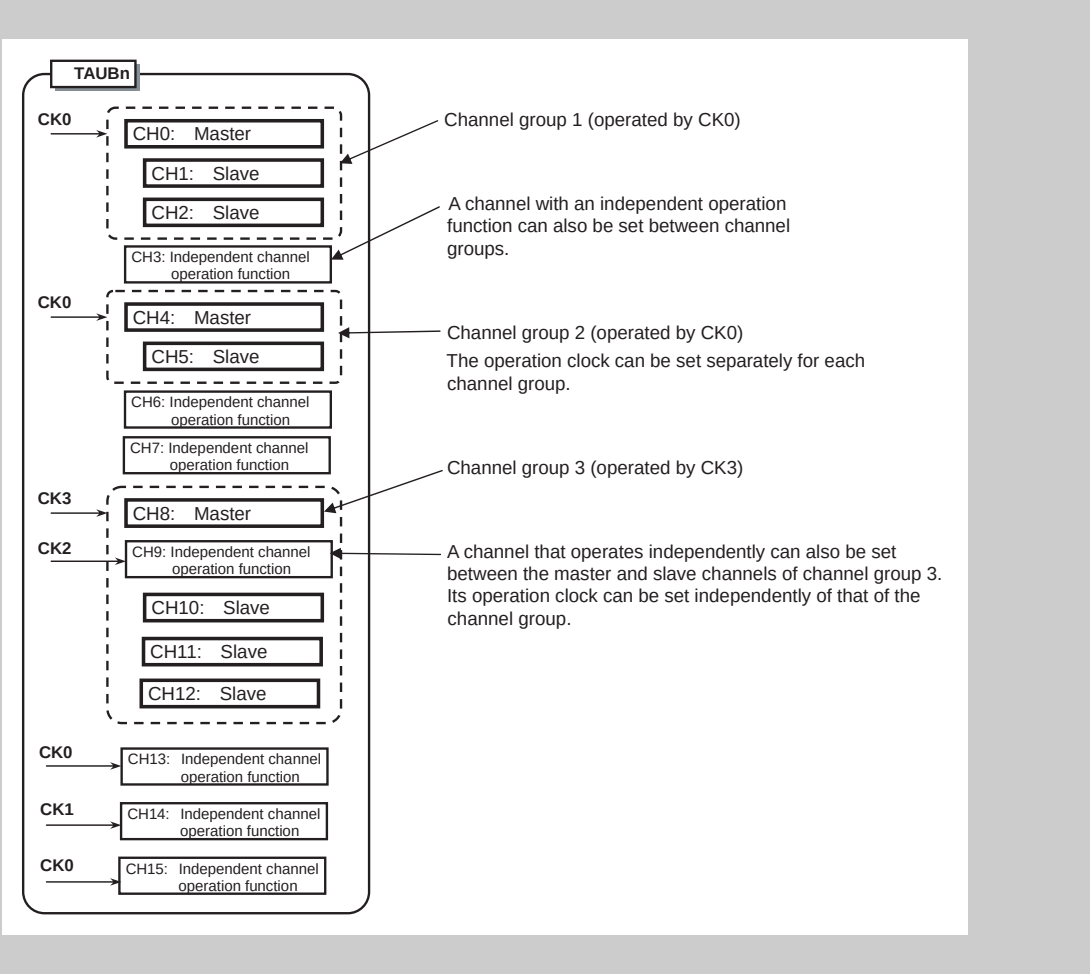


Figure 16-3 Grouping of the channels and assignment of operation clocks

INTTAUBnIm, start trigger, and count clock

- Master channels can transfer an interrupt request (INTTAUBnI), the start trigger, and the count clock to slave channels.
- Slave channels can use INTTAUBnI, the start trigger, and the count clock of the master channels but cannot transfer their INTTAUBnI, start trigger, or count clock to the lower channels.
- A master channel cannot use INTTAUBnI, the start trigger, or the count clock of the higher master channels.

16.6.2 Simultaneous start and stop of synchronous channel counters

Channels that are operated synchronously can be started and stopped simultaneously, both within a TAUB unit, and between TAUB units.

(1) Simultaneous start and stop within a TAUB unit

- To simultaneously start synchronized channels, the TAUBnTS.TSm bits of the channels must be set at the same time.
- To simultaneously stop synchronized channels, the TAUBnTT.TTm bits of the channels must be set at the same time.

Writing to the TAUBnTS.TSm bits sets the corresponding TAUBnTE.TEm bits to 1, enabling counting. TAUBnTS.TSm = 1 only enables the corresponding counter to start; the exact time that it starts depends on the operation mode.

(2) Simultaneous start between TAUB units

Counters in different TAUB units can also be started simultaneously if the corresponding counters are enabled before receiving the simultaneous trigger signal. The simultaneous start trigger register is then sent to the TAUBnTSSTm input.

16.7 Simultaneous Rewrite

16.7.1 Introduction

Simultaneous rewrite describes the ability to change the compare/start value and the output logic of multiple channels at the same time.

The corresponding data and control registers (TAUBnCDRm and TAUBnTOLm) can nevertheless be written at any time. The new value does not affect the counter operation or the output signal until simultaneous rewrite is triggered.

Simultaneous rewrite can be triggered by:

- The counter on the master channel or upper channel (depending on the selected operation mode) reaching a certain value
- INTTAUBnI being issued on the upper channel specified by TAUBnRDC.RDCm

There are four methods for simultaneous rewrite. These are listed in the following table, along with how to specify them and when they cause simultaneous rewrite to be triggered.

Table 16-8 Simultaneous rewrite methods and when they are triggered

Method	Simultaneous rewrite triggered when	TAUBn RDE. RDEm	TAUBn RDS. RDSm	TAUBn RDM. RDMm
-	No simultaneous rewrite	0	0	0
A	The master channel (re)starts counting	1	0	0
B	The slave channel starts counting down at the upper peak of a triangular cycle	1	0	1
C1	INTTAUBnIm is generated on an upper channel specified by TAUBnRDC.RDCm	1	1	0
TOLm	For TOLm, the following table shows whether TOLm can be rewritten during operation. The TOLm rewrite method is the same as that of CDRn.			

The following table lists which of these four methods is available for each channel operation function. For details about the individual channel operation functions, see the corresponding sections in *16.13 “Independent Channel Operation Functions” on page 979* and *16.18 “Synchronous Channel Operation Functions” on page 1070*.

Table 16-9 Channel operation functions and methods they use (1/2)

Function	A	B	C1	TOLm
Simultaneous rewrite trigger output function type 1			X	
PWM output function	X		X	X
One-shot pulse output function	X			
Delay pulse output function	X			
Triangle PWM output function		X	X	X

Table 16-9 Channel operation functions and methods they use (2/2)

Function	A	B	C1	TOLm
Triangle PWM output function with dead time		X	X	
AD conversion trigger output function type 1	X		X	
AD conversion trigger output function type 2		X	X	

16.7.2 How to control simultaneous rewrite

The following figure shows the general procedure for simultaneous rewrite. The three main blocks (Initial settings, Start counter & count operation, and Simultaneous rewrite) are explained afterwards.

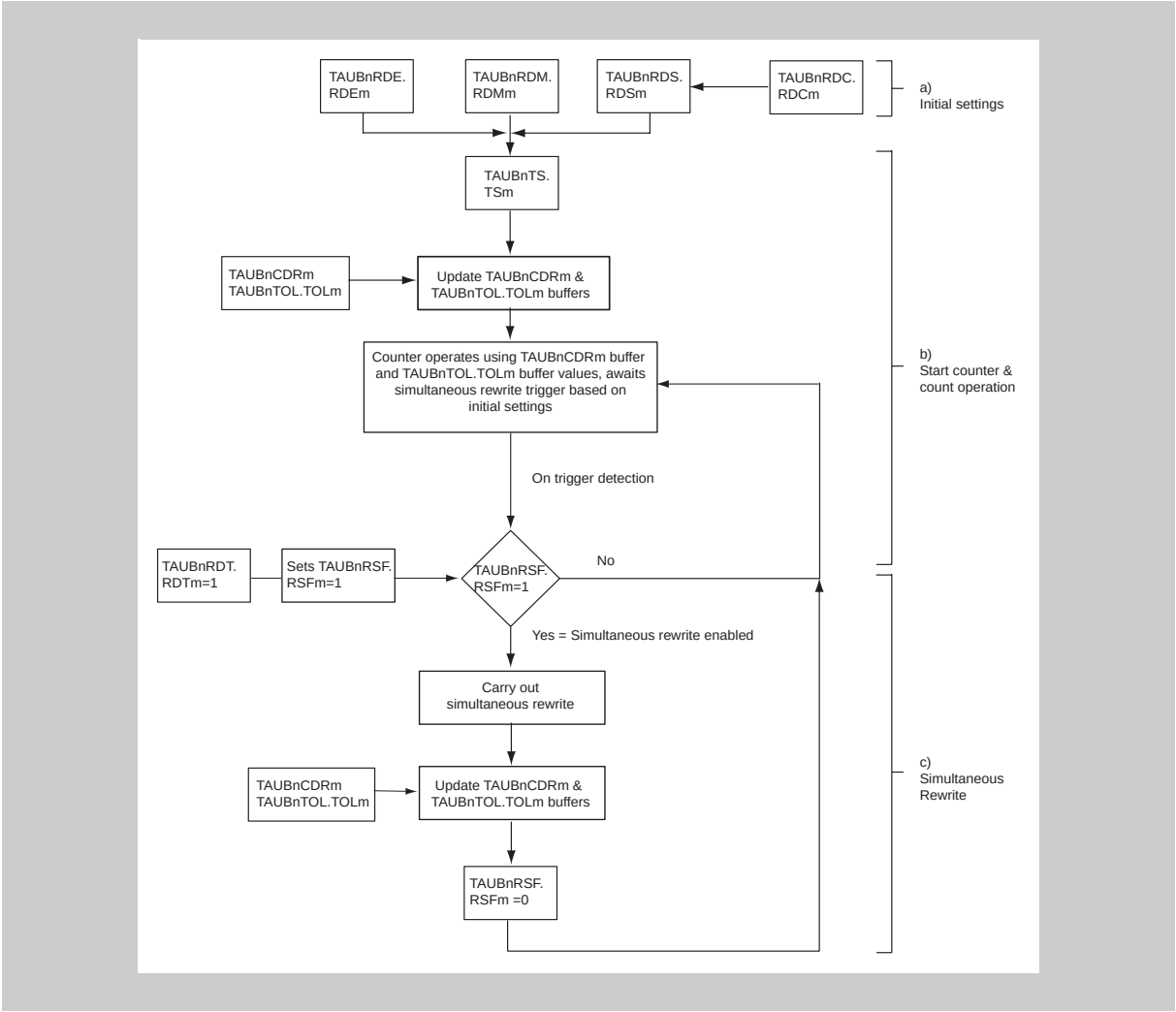


Figure 16-4 General procedure for simultaneous rewrite

(1) Initial settings

- To enable simultaneous rewrite in channel m, set TAUBnRDE.RDEm = 1
- To select the type of simultaneous rewrite, set TAUBnRDM.RDMm and TAUBnRDS.RDSm according to the values in *Table 16-8 “Simultaneous rewrite methods and when they are triggered” on page 952*
- To select which upper channel is monitored for the simultaneous rewrite trigger use TAUBnRDC.RDCm (prerequisite: TAUBnRDS.RDSm is set to upper channel)

(2) Start counter and count operation

- To start all the TAUBnCNTm counters in the channel group, set the corresponding TAUBnTS.TSm bits to 1. TAUBnTOL.TOLm and the values in the data registers (TAUBnCDRm) are written to the corresponding TAUBnTOL.TOLm buffer (TAUBnTOL.TOLm buf) and data buffer registers (TAUBnCDRm buf) and the counters start.
- Setting the reload data trigger bit (TAUBnRDT.RDTm) to 1 sets the reload flag (TAUBnRSF.RSFm) to 1, enabling simultaneous rewrite. TAUBnRDT.RDTm then immediately returns to 0, but TAUBnRSF.RSFm remains at 1 until simultaneous rewrite has taken place.
- When the specified trigger for simultaneous rewrite is detected, the TAUBnRSF.RSFm bit is checked to see if simultaneous rewrite is enabled (TAUBnRSF.RSFm = 1). If it is, simultaneous rewrite is carried out. Otherwise the value of the TAUBnRSF.RSFm bit is re-evaluated the next time the trigger is detected.

(3) Simultaneous rewrite

- When the simultaneous rewrite trigger is detected and simultaneous rewrite is enabled (TAUBnRSF.RSFm = 1), the current values of the data registers are copied to their buffers. These values are then written to the corresponding counters and the values are applied the next time the counter starts or restarts.
- The TAUBnRSF.RSFm bit is set to 0, and the system awaits the next simultaneous rewrite trigger.

16.7.3 Other general rules of simultaneous rewrite

The following rules also apply:

- TAUBnRDE.RDEm, TAUBnRDS.RDSm, TAUBnRDM.RDMm, and TAUBnRDC.RDCm cannot be changed while the counter is in operation (TAUBnTE.TEm = 1).
- TAUBnTOL.TOLm can only be rewritten during operation when in PWM output function or triangle PWM output function. For all other output functions, TAUBnTOL.TOLm must be written before the counter starts. If it is rewritten in another function, TAUBnTTOUTm outputs an invalid wave.
- When an upper channel is used as the channel issuing the simultaneous rewrite trigger (TAUBnRDS.RDSm = 1), the TAUBnRDC.RDCm bit controls all the lower channels. This means that if the TAUBnRDC.RDCm bits of CH2 and CH7 are set to 1 and the TAUBnRDC.RDCm bits of other channels are set to 0, CH2 and CH7 serve as simultaneous rewrite trigger generation channels. CH2 controls the lower channels CH3 to CH6, and CH7 controls the lower channels CH8 to CH15.
- If simultaneous rewrite is enabled and an upper channel is selected for the simultaneous rewrite trigger (TAUBnRDE.RDEm and TAUBnRDS.RDSm = 1) but no upper channel is set (TAUBnRDC.RDC[15:0] = 0), simultaneous rewrite cannot take place.

16.7.4 Types of simultaneous rewrite

In the following section the four simultaneous rewrite methods are explained using timing diagrams.

(1) Simultaneous rewrite when the master channel (re)starts counting (method A)

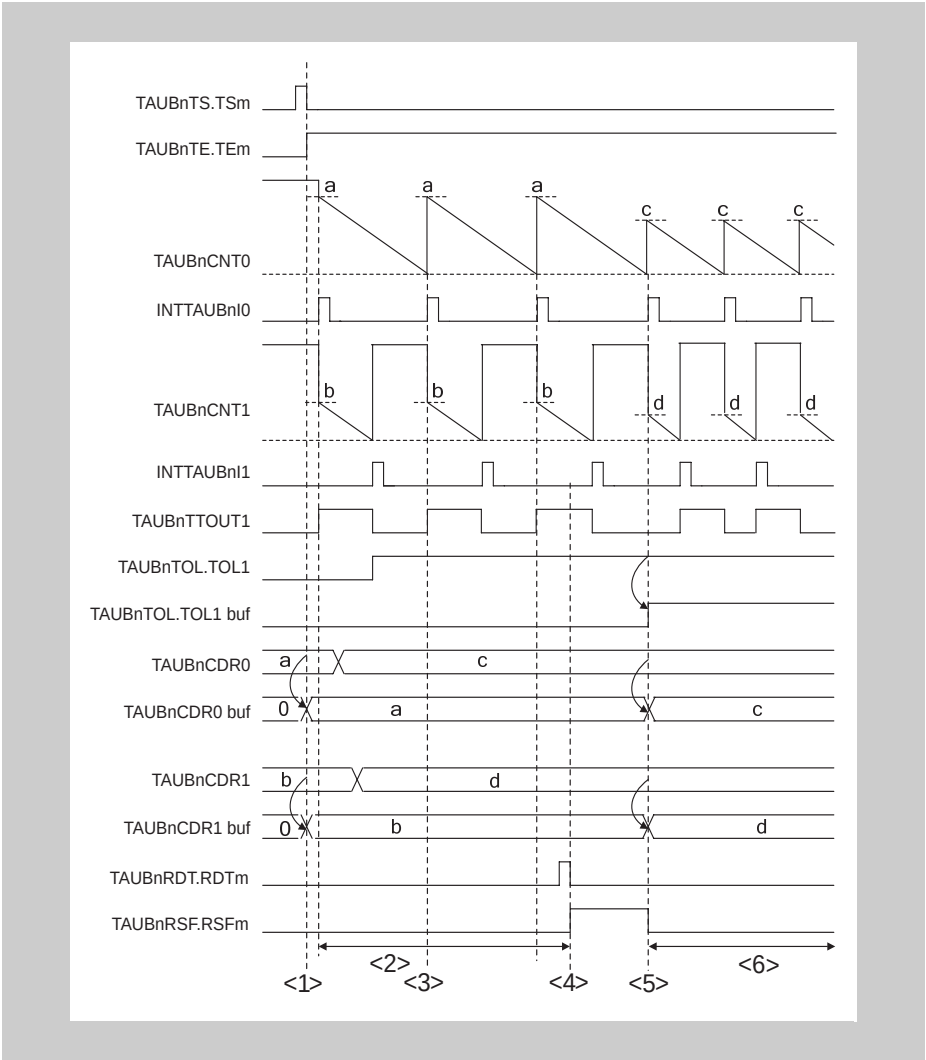


Figure 16-5 Simultaneous rewrite when the master channel (re)starts counting

Setup:

- CH0 is the master channel, counting down, CH1 represents an arbitrary slave channel, and simultaneous rewrite method A is applied.

Description:

1. When the counter starts, the value of TAUBnCDRm is copied to the TAUBnCDRm buffer and the value of TAUBnTOL.TOLm is copied to the TAUBnTOL.TOLm buffer. The TAUBnCDRm buffer value is written to the counter.
2. The TAUBnCDRm and TAUBnTOL.TOLm registers can be written at any time, but the values do not affect the counter as the counter reads the buffer values.
3. CH0 restarts counting, but simultaneous rewrite does not occur because it is disabled (TAUBnRSF.RSFm = 0).
4. The reload data trigger bit (TAUBnRDT.RDTm) is set to 1 which sets the status flag (TAUBnRSF.RSFm = 1), enabling simultaneous rewrite.
5. Simultaneous rewrite is triggered by counter TAUBnCNT0 starting to count down. The TAUBnCDRm value is written to the TAUBnCDRm buffer and the TAUBnTOL.TOLm value is written to the TAUBnTOL.TOLm buffer.
   – The counter starts to count down from the value in the TAUBnCDRm buffer and the TAUBnRSF.RSFm bit is reset to 0.
   – The output logic specified by TAUBnTOL.TOLm becomes effective.
6. The counters count down and await the next simultaneous rewrite trigger. The values of TAUBnCDRm and TAUBnTOL.TOLm can be changed again.

(2) Simultaneous rewrite at the peak of a triangular cycle of the slave channel (method B)

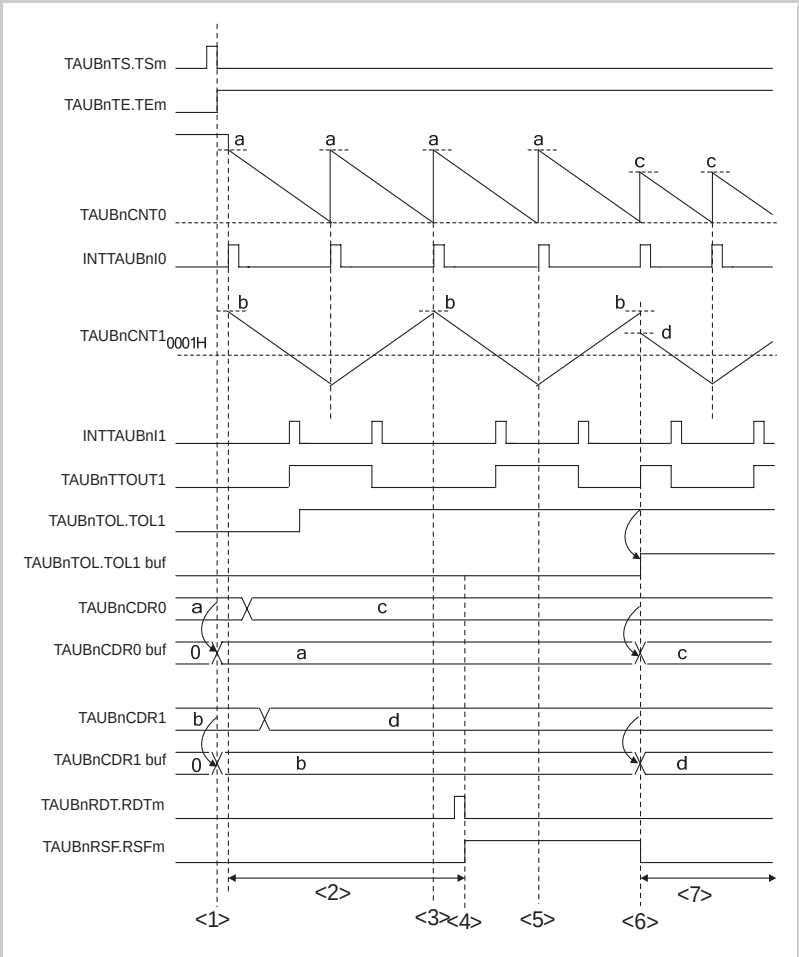


Figure 16-6 Simultaneous rewrite at the peak of a triangular cycle of the slave channel

Setup:

- CH0 is the master channel, counting up and down, CH1 represents an arbitrary slave channel, and simultaneous rewrite method B is applied.

Description:

1. When the counter starts, the value of TAUBnCDRm is copied to the TAUBnCDRm buffer. The buffer value is written to the counter.
2. The TAUBnCDRm and TAUBnTOL registers can be written at any time, but the values do not affect the counter as the counter reads the buffer values.
3. Simultaneous rewrite does not occur because it is disabled (TAUBnRSF.RSFm = 0).
4. The reload data trigger bit (TAUBnRDT.RDTm) is set to 1 which sets the status flag (TAUBnRSF.RSFm = 1), enabling simultaneous rewrite.
5. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by the slave channel starting to count down at an upper peak.
6. Simultaneous rewrite is triggered; the TAUBnCDRm value is written to the TAUBnCDRm buffer, the TAUBnTOL.TOLm value is written to the TAUBnTOL.TOLm buffer.
   – The counters start to count down from the value in the TAUBnCDRm buffer and the TAUBnRSF.RSFm bit is reset to 0.
   – The output logic specified by TAUBnTOL.TOLm becomes effective.
7. The counters count down and await the next simultaneous rewrite trigger. The values of TAUBnCDRm and TAUBnTOL.TOLm can be changed again.

(3) Simultaneous rewrite when INTTAUBnIm is generated on an upper channel specified by TAUBnRDC.RDCm (method C1)

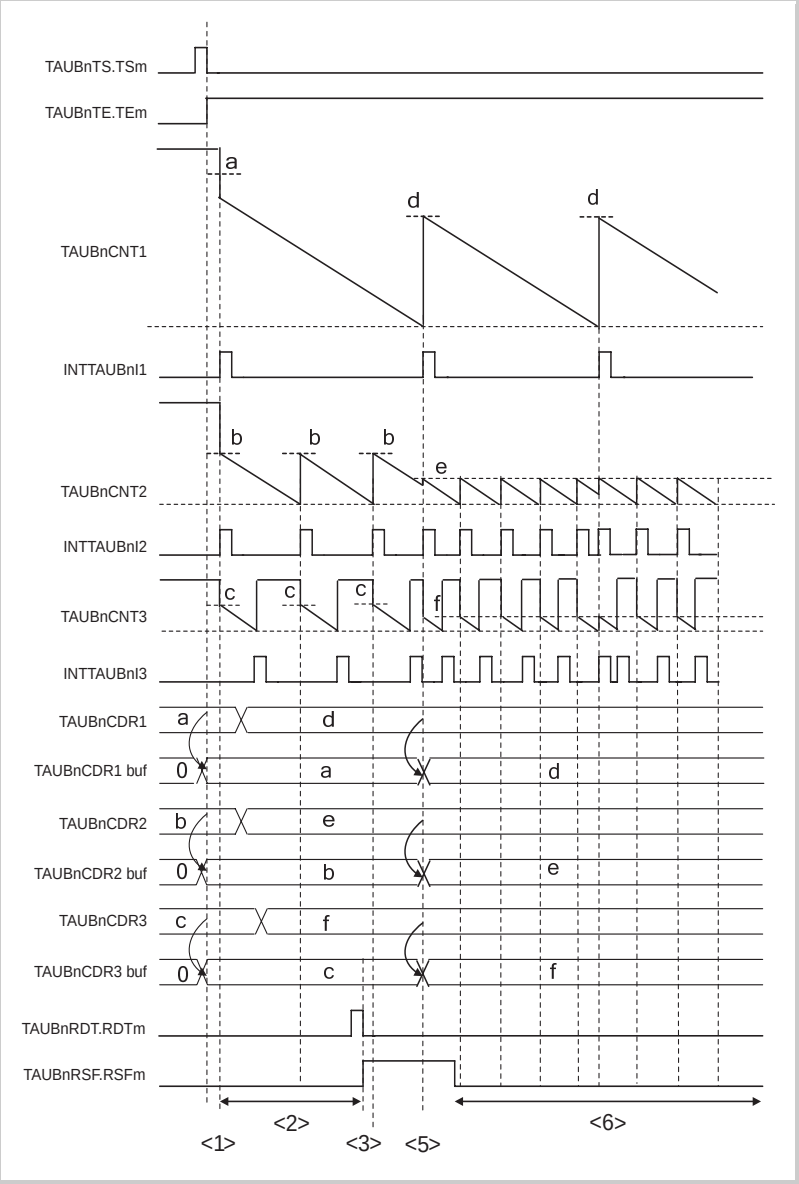


Figure 16-7 Simultaneous rewrite when INTTAUBnIm is generated on an upper channel specified by TAUBnRDC.RDCm

Setup:

- CH1 is an upper channel, counting down. CH2 is a master channel, CH3 is the slave channel, and simultaneous rewrite method C1 is applied. The TAUBnRDC register specifies which upper channel is monitored for an INTTAUBnI trigger.

Description:

1. When the counter starts, the value of TAUBnCDRm is copied to the TAUBnCDRm buffer. The buffer value is written to the counter.
2. The TAUBnCDRm register can be written at any time, but the value does not affect the counter as the counter reads the buffer value.
3. The reload data trigger bit (TAUBnRDT.RDTm) is set to 1 which sets the status flag (TAUBnRSF.RSFm = 1), enabling simultaneous rewrite.
4. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by an interrupt on channel 1.
5. Simultaneous rewrite is triggered by INT1 which is caused by counter 1 reaching 0000_H. The TAUBnCDRm values are written to the corresponding TAUBnCDRm buffers, the counters start to count down from the values in the TAUBnCDRm buffers, and the TAUBnRSF.RSFm bit is reset to 0.
6. The counter counts down and awaits the next simultaneous rewrite trigger. The values of the TAUBnCDRm registers can be changed again.

16.8 Channel Output Modes

The output of the TAUBnTTOUTm pin can be controlled in two ways, the latter of which can be further split into individual modes:

- By software (TAUBnTOE.TOEm = 0)

  When controlled by software, the output register bit (TAUBnTO.TOm) can be written and the value of the bit is transferred to the output pin (TAUBnTTOUTm).

- By TAUB signals (TAUBnTOE.TOEm = 1)

  When operated by TAUB signals, the output level of TAUBnTTOUTm is set or reset or toggled by internal signals. The value of TAUBnTO.TOm is updated accordingly to reflect the value of TAUBnTTOUTm.

  – Independently (TAUBnTOM.TOMm = 0)

    When operated independently, the output of the TAUBnTTOUTm pin is only affected by settings of channel m. Therefore, independent channel operation must be selected (TAUBnTOM.TOMm = 0).

  – Synchronously (TAUBnTOM.TOMm = 1)

    When operated synchronously, the output of the TAUBnTTOUTm pin is affected by settings of channel m and those of other channels. Therefore, synchronous channel operation must be selected for all participating channels (TAUBnTOM.TOMm = 1).

The TAUBnTO.TOm bit can always be read to determine the current value of TAUBnTTOUTm, regardless of whether the pin is controlled by software, operated independently, or operated synchronously.

Control bits

The settings of the control bits required to select a specific channel output mode are listed in *Table 16-10 “Channel output modes” on page 963*.

The channel output modes are described in detail in

- *16.8.2 “Channel output modes controlled independently by TAUBn signals” on page 965*
- *16.8.3 “Channel output modes controlled synchronously by TAUBn signals” on page 966*.

Output logic

Positive logic or inverted logic of the output is specified by control bit TAUBnTOL.TOLm.

The value of the TAUBnTOL.TOLm bit must be set before the counter is started. It can only be changed during operation in PWM output function or triangle PWM output function. Otherwise, changes to TAUBnTOL.TOLm result in an invalid TAUBnTTOUTm signal.

See *16.7 “Simultaneous Rewrite” on page 952*.

The various channel output modes and the channel output control bits are listed in the following table.

Multiple outputs

For a function on channel m that uses its output and the outputs of other channels (q):

- If channel m requires a certain operation mode for channel q, set the operation mode on channel q.

- If channel m does not require a certain operation mode for channel q, the counter on channel q must be disabled (TAUBnTE.TEq = 0), i.e. no operation mode can be used on channel q, even one that does not generate an output.

Table 16-10 Channel output modes

Channel output mode	TAUBn TOE. TOEm	TAUBn TOM. TOMm	TAUBn TOC. TOCm	TAUBn TDE. TDEm
By software				
Independent channel output mode controlled by software	0	x		
By TAUB signals, independently				
Independent channel output mode 1	1	0	0	0
Independent channel output mode 2			1	
By TAUB signals, synchronously				
Synchronous channel output mode 1	1	1	0	0
Synchronous channel output mode 2			1	
with dead time output				1

- All combinations not listed in this table are forbidden.
- Bits marked with an x can be set to any value.

Note The following bits cannot be changed during count operation (TAUBnTE.TE = 1):

- TAUBnTOE.TOEm,
- TAUBnTOM.TOMm,
- TAUBnTOC.TOCm,
- TAUBnTDE.TDEm

16.8.1 General procedure for specifying a channel output mode

The following steps describe the general procedure for specifying a TAUBnTTOUTm channel output mode. The prerequisite is that timer output operation is disabled (TAUBnTOE.TOEm = 0).

1. Set TAUBnTO.TOm to specify the initial level of the TAUBnTTOUTm output.
2. Set the channel output mode using *Table 16-10 "Channel output modes" on page 963* and the output logic using the TAUBnTOL.TOLm bit.
3. Start the counter (TAUBnTS.TSm = 1).

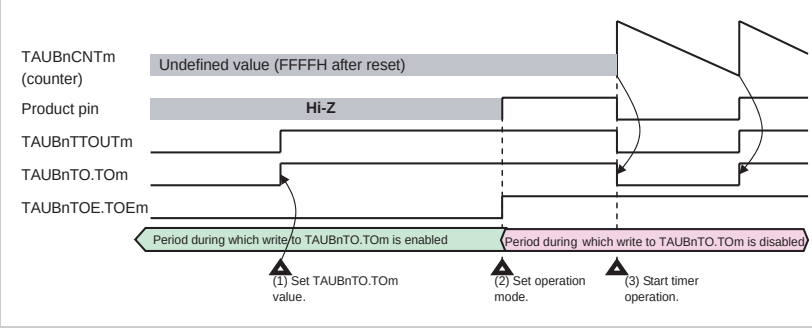


Figure 16-8 General procedure for specifying a TAUBnTTOUTm channel output mode

The following figure shows a general illustration of how the output changes when the counter is enabled:

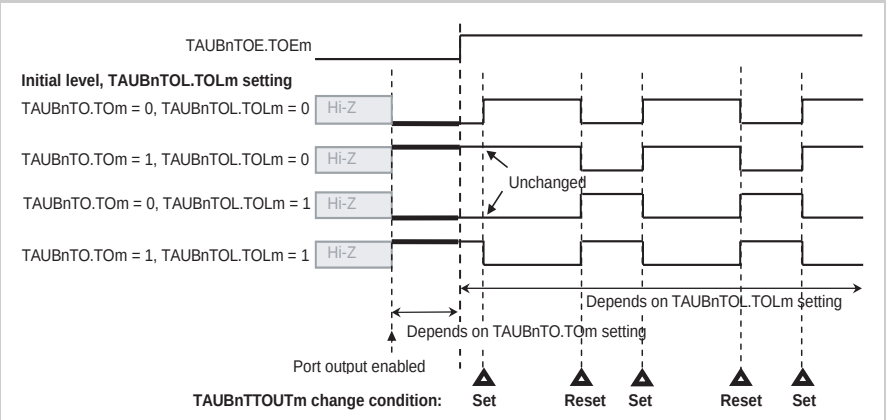


Figure 16-9 General change of the TAUBnTTOUTm output

- TAUBnTO.TOm sets the initial value of TAUBnTTOUTm and can be changed while TAUBnTOE.TOEm = 0.
- TAUBnTOL.TOLm specifies whether the set signal sets TAUBnTO.TOm to high (TAUBnTOL.TOLm = 0) or low (inverted logic, TAUBnTOL.TOLm = 1).

16.8.2 Channel output modes controlled independently by TAUBn signals

This chapter lists the channel output modes that are controlled independently by TAUBn signals. The control bits used to specify a mode are listed in *Table 16-10 “Channel output modes” on page 963*.

(1) Independent channel output mode 1

Set/reset conditions In this output mode, TAUBnTTOUTm toggles when INTTAUBnIm is detected. The value of TAUBnTOL.TOLm is ignored.

Prerequisites None, other than those in *Table 16-10 “Channel output modes” on page 963*.

(2) Independent channel output mode 2

Set/reset conditions In this output mode, TAUBnTTOUTm is set when INTTAUBnIm occurs upon count start and reset when INTTAUBnIm occurs due to a match between TAUBnCNTm and TAUBnCDRm.

Prerequisites None, other than those in *Table 16-10 “Channel output modes” on page 963*.

16.8.3 Channel output modes controlled synchronously by TAUBn signals

This chapter lists the channel output modes that are controlled synchronously by TAUBn signals. The control bits used to specify a mode are listed in *Table 16-10 “Channel output modes” on page 963*.

(1) Synchronous channel output mode 1

Set/reset conditions In this output mode, INTTAUBnIm of the master channel serves as the set signal and INTTAUBnIm of the slave channel as the reset signal. If INTTAUBnIm of the master channel and INTTAUBnIm of the slave channel are generated at the same time, INTTAUBnIm of the slave channel (reset signal) has priority over INTTAUBnIm (set signal) of the master channel, i.e. the master channel is ignored.

Prerequisites None, other than those in *Table 16-10 “Channel output modes” on page 963*.

(2) Synchronous channel output mode 2

In this output mode, the operation mode must be set to up/down count mode. The result is a triangle PWM wave at TAUBnTTOUTm. For details, see *16.21.1 "Triangle PWM output function" on page 1114*.

Set/reset conditions TAUBnCNTm of the slave channel counts down and up alternatively. When it passes 0001_H it generates an interrupt, causing TAUBnTTOUTm to toggle.

Prerequisites A set of two channels is required to generate the triangle PWM output. TAUBnTTOUTm must be set to 0 before the function starts.

(3) Synchronous channel output mode 2 with dead time output

In this output mode, a dead time delay is added to TAUBnTTOUTm. The set/reset conditions are shown in the following figure.

Set/reset conditions

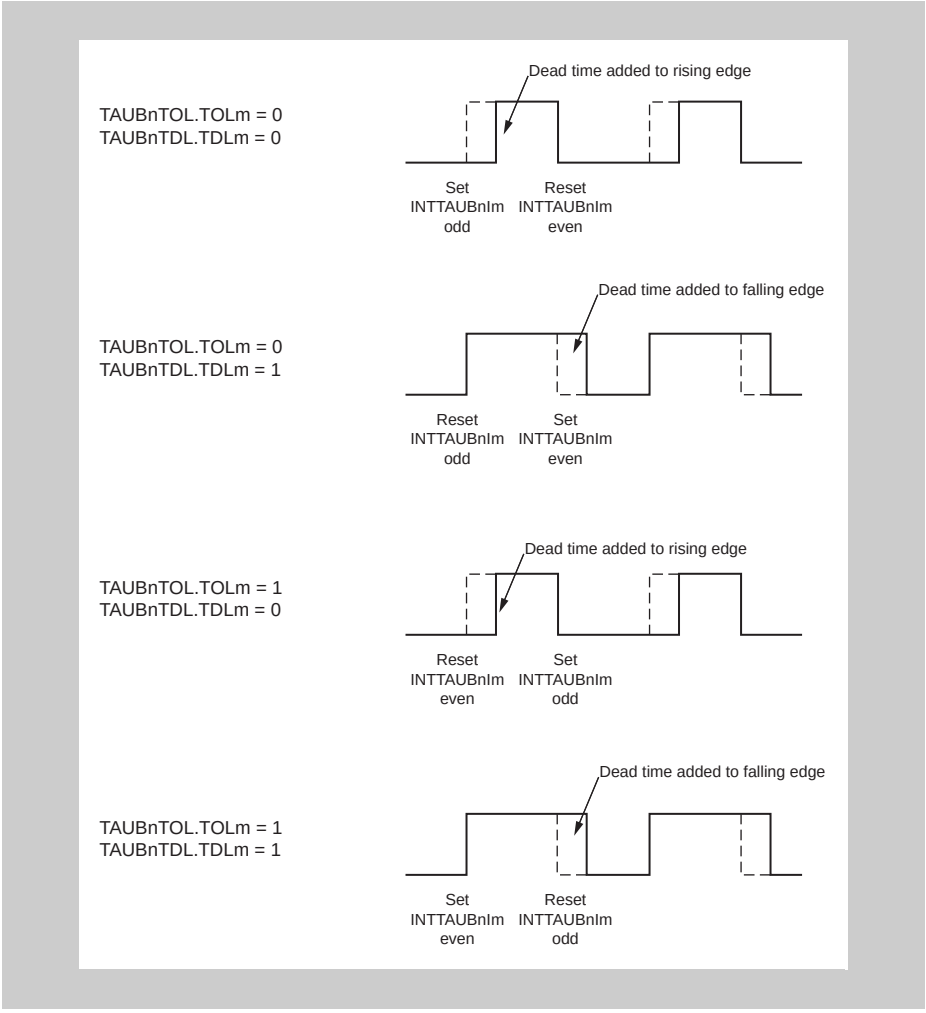


Figure 16-10 Set/reset conditions for synchronous channel output mode 2 with dead time output

The edge to which the dead time is added is specified using the TAUBnTDL.TDLm bit; for rising edge set TAUBnTDL.TDLm = 0 and for falling edge set TAUBnTDL.TDLm = 1.

Prerequisites Dead time control requires a set of three channels, each operating in the following modes:

- One master channel

  The master channel must be set to interval timer mode

- One even slave channel

  The even slave channel must be set to up/down count mode

- One odd slave channel (even channel + 1)

  The odd slave channel must be set to one-count mode

The values of the following bits must be the same for the odd channel and the even channel:

- TAUBnTOE.TOEm,
- TAUBnTOM.TOMm,
- TAUBnTOC.TOCm,
- TAUBnTDE.TDEm

16.9 Start Timing of Operating Modes

This chapter describes when the counters of the different operating modes start after the TAUBnTS.TSm bit is set to 1.

In all modes, the value of the data register and whether or not an interrupt is issued depends on the individual mode and corresponding register settings.

16.9.1 Interval timer mode, judge mode, capture mode, and up/down count mode

The counter starts at the start of the next count clock cycle after TAUBnTS.TSm is set to 1. The value of data register is also loaded at the point the counter starts.

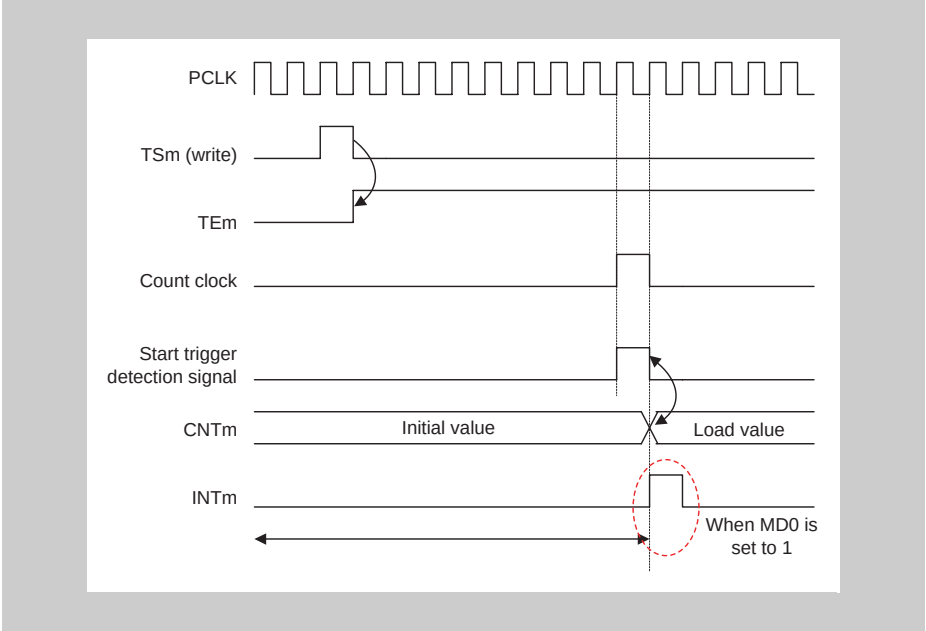


Figure 16-11 Start timing of interval timer mode, judge mode, capture mode, and up/down count mode

Note In up/down count mode, MD0 must be set to 0.

16.9.2 Event count mode

The value of the data register is loaded as soon as TAUBnTS.TSm is set to 1. The counter also starts immediately. The value of the data register changes with the subsequent count clock cycles.

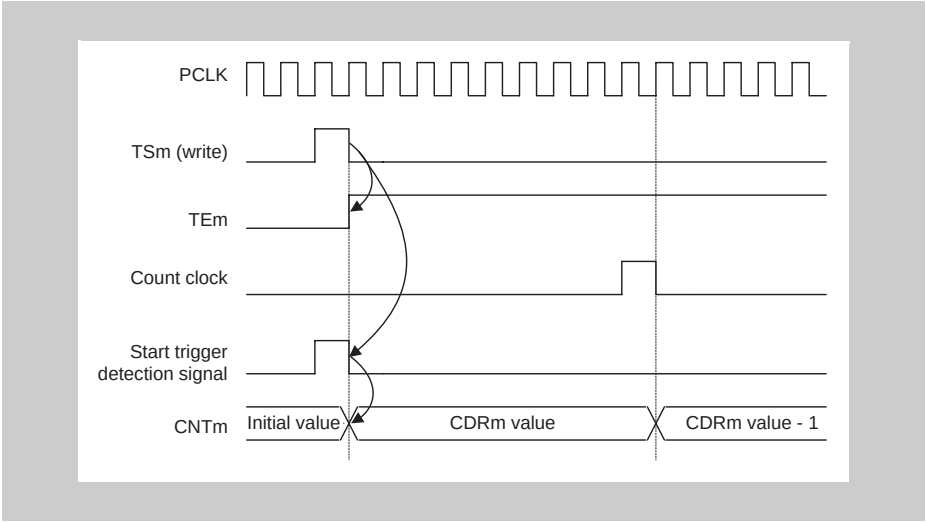


Figure 16-12 Start timing of event count mode

16.9.3 All other operating modes

In all other operating modes, the count clock cycles are ignored with regard to starting the counter. The counter is only triggered by detection of a valid TAUBnTTINm edge. The value of data register is also loaded at the point the counter starts. Nevertheless, the count clock cycles determine the frequency with which all operations take place.

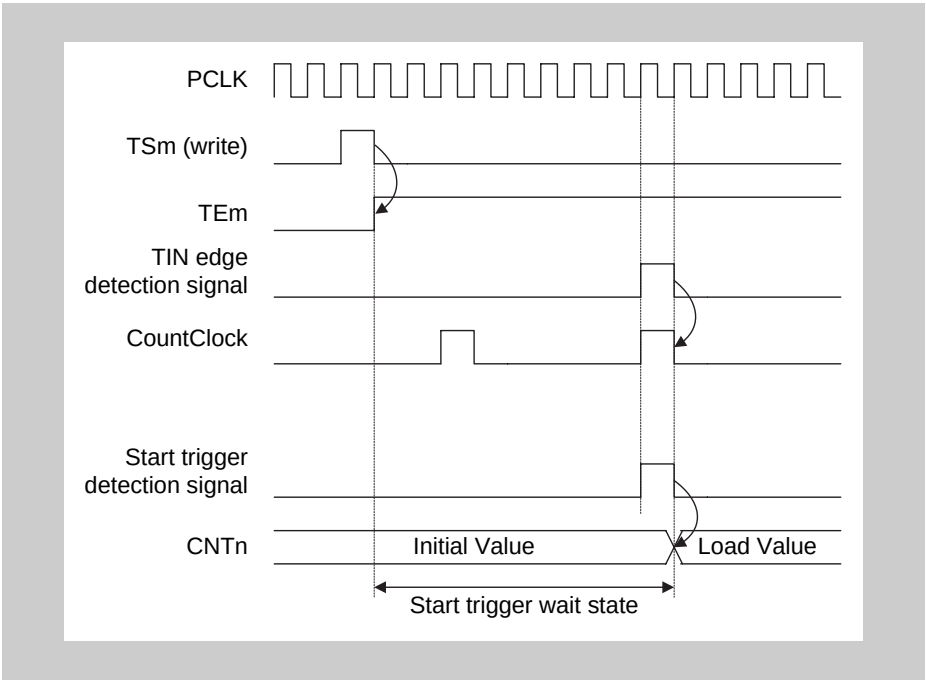


Figure 16-13 Start timing of all other operating modes

16.10 TAUBnTTOUTm Toggle and INTTAUBnIm Generation When Counter Start Is Triggered (MD0 Bit)

It is possible to specify whether an INTTAUBnIm is generated when the counter starts, gets restarted or is triggered by an external signal, using the TAUBnCMOR.MD0 bit. The effect of the bit depends on the selected mode, as shown in the following table. The effects of INTTAUBnIm on TAUBnTTOUTm depend on the selected channel operation function.

Table 16-11 Effect of CMOR.MD0 bit on generation of INTTAUBnIm when counter is triggered

Mode	TAUBnCMOR.MD0 bit	INTTAUBnIm generated when counter is (re)started or triggered by TINm input signal
Interval timer mode Capture mode Count capture mode	0	No
	1	Yes
Capture & one-count mode Capture & gate count mode Event count mode Up/down count mode	0	No
One-count mode Gate count mode	0/1	No, regardless of setting of TAUBnCMOR.MD0 bit.
Pulse one-count mode		Yes, regardless of setting of TAUBnCMOR.MD0 bit.

Note As an example see *Figure 16-30 "Forced restart operation, TAUBnCMORm.MD0 = 1"* and *Figure 16-31 "Forced restart operation, TAUBnCMORm.MD0 = 0"*.
See the description of *"Role of the MD0 bit"* also.

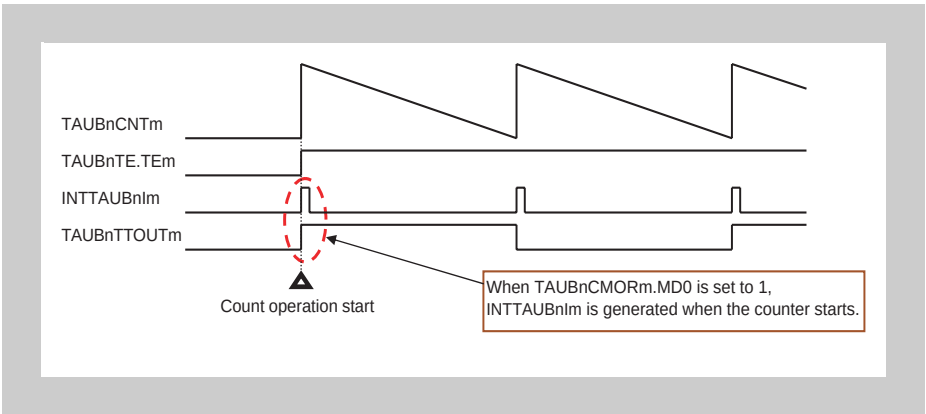


Figure 16-14 INTTAUBnIm generated when counter starts

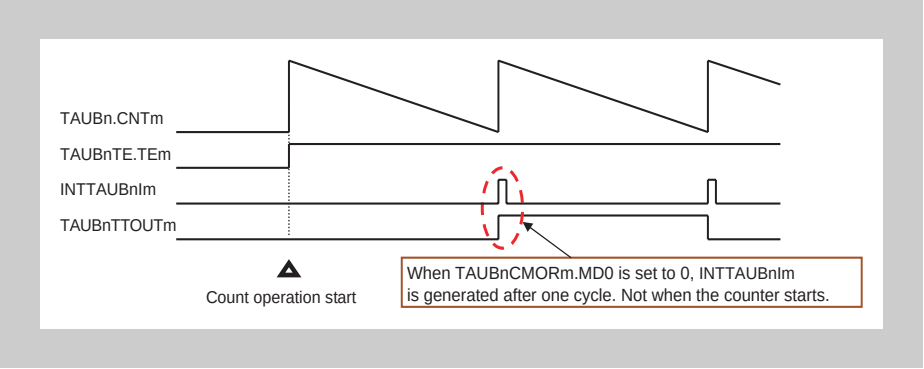


Figure 16-15 INTTAUBnIm not generated when counter starts

16.11 Interrupt Generation upon Overflow

Certain independent functions that count up, overflow without generating an interrupt when they reach $FFFF_H$. This section describes how it is possible to generate an interrupt, by combining a channel operating in one of these modes with a channel in a different operation mode which counts down.

The appropriate operation mode for the second channel depends on the operation mode of the first channel. Nevertheless, the principle is the same for all combinations:

- Find a operation mode for the second channel that counts down in such a manner, that it reaches 0000_H at the same time as the first channel overflows (TAUBnCNTm = $FFFF_H$).
- Set TAUBnCDRm of the second channel to $FFFF_H$
- The two channels must count at the same speed (i.e. they must have the same count clock)
- Both channels are triggered by the same TAUBnTTINm input

Result: The down-counter of the second channel reaches 0000_H at exactly the same time as the up-counter of the first channel overflows (TAUBnCNTm = $FFFF_H$). Thus the second channel generates the desired interrupt.

The following sections list the operating modes that count down that are required to match specific operating modes that count up, as well as example timing diagrams.

16.11.1 Capture mode

Applies to • TAUBnTTINm input pulse interval measurement function

Combine with Interval timer mode

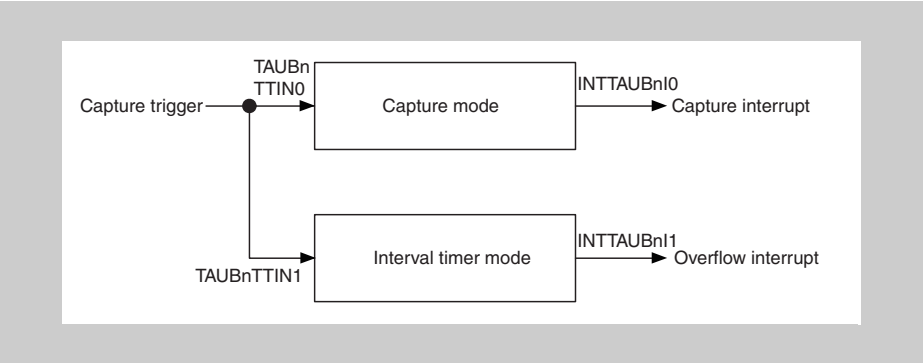


Figure 16-16 Combination of capture mode and interval timer mode

Timing diagram

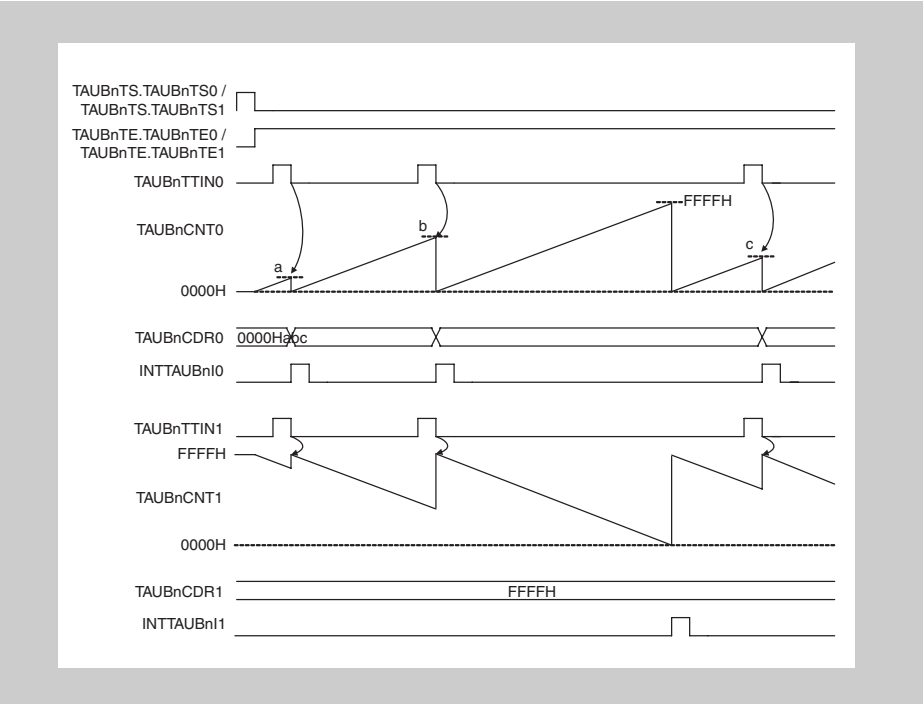


Figure 16-17 Interrupt generation via combination of capture mode and interval timer mode

16.11.2 Capture & one-count mode

Applies to
- TAUBnTTINm input signal width measurement function

Combine with
One-count mode

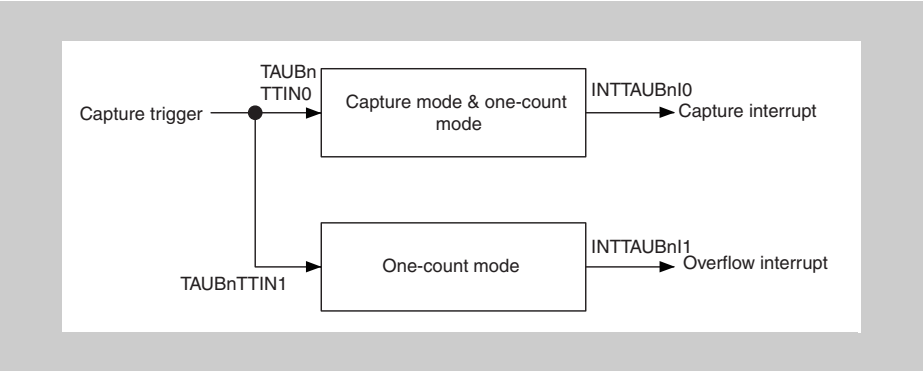


Figure 16-18 Combination of capture & one-count mode and one-count mode

Timing diagram

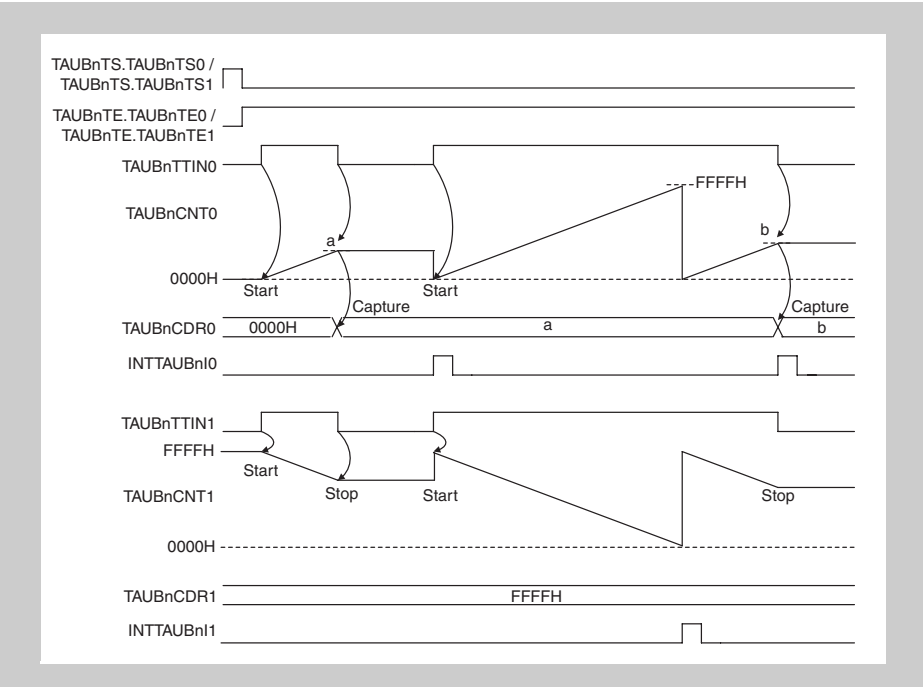


Figure 16-19 Interrupt generation via combination of capture & one-count mode and one-count mode

16.11.3 Count capture mode

Applies to
- TAUBnTTINm input position detection function

Combine with
Interval timer mode

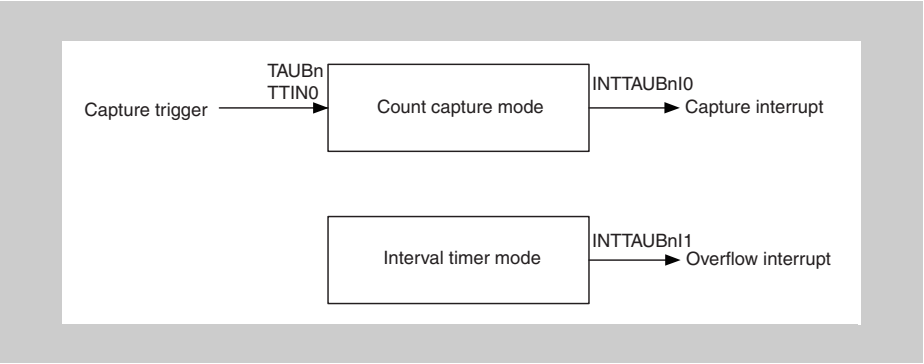


Figure 16-20 Combination of count capture mode and interval timer mode

Timing diagram

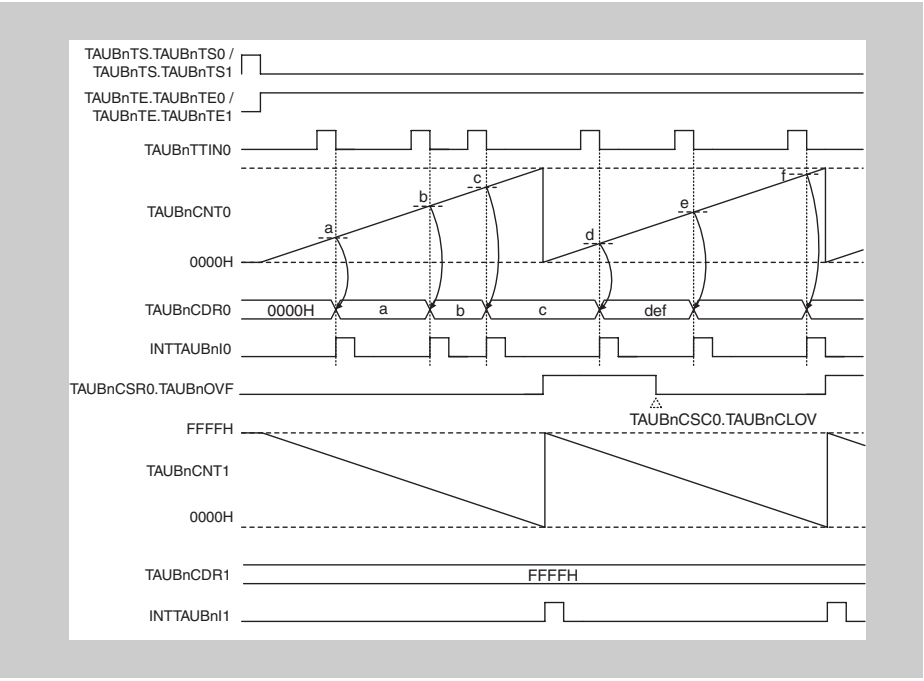


Figure 16-21 Interrupt generation via combination of count capture mode and interval timer mode

In the above timing diagram, TAUBnCSRm.TAUBnOVF is set to 1 when TAUBnCNTm overflows.

TAUBnCSRm.TAUBnOVF is cleared by writing 1 to TAUBnCSCm.TAUBnCLOV.

16.11.4 Capture & gate count mode

Applies to
- TAUBnTTINm input period count detection function

Combine with
Gate count mode

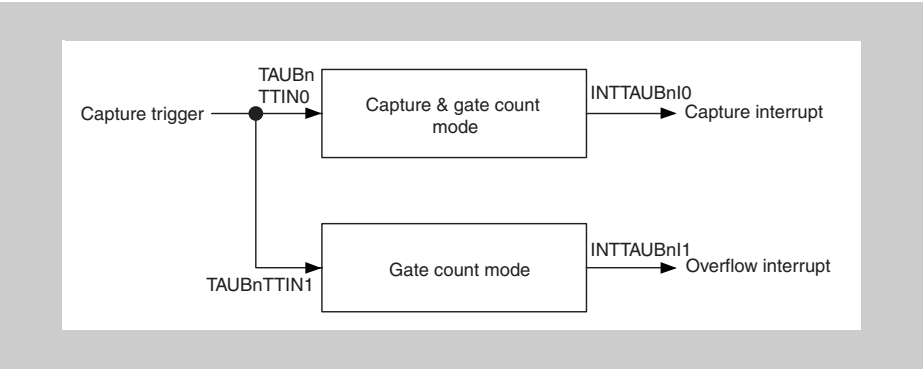


Figure 16-22 Combination of capture & gate count mode and gate count mode

Timing diagram

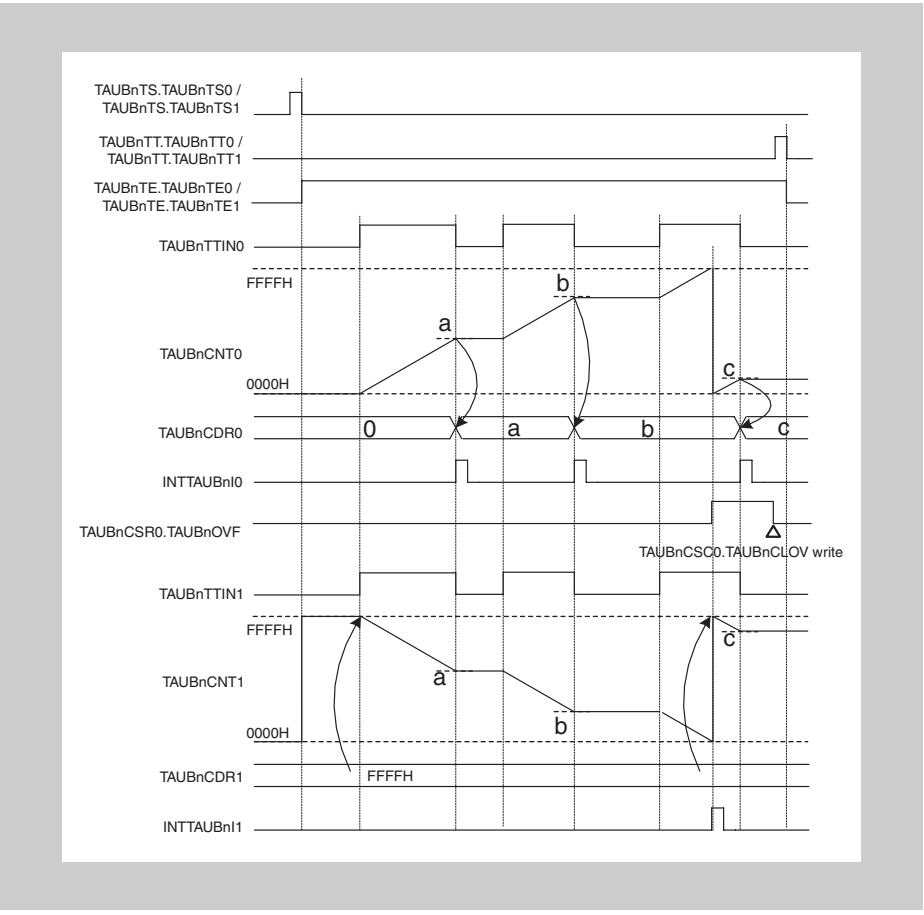


Figure 16-23 Interrupt generation via combination of capture & gate count mode and gate count mode

In the above timing diagram, TAUBnCSRm.TAUBnOVF is set to 1 when TAUBnCNTm overflows. TAUBnCSRm.TAUBnOVF is cleared by writing 1 to TAUBnCSCm.TAUBnCLOV.

16.12 TAUBnTTINm Edge Detection

Edge detection is based on the operation clock. This means that an edge can only be detected at the next rising edge of the operation clock. This can lead to a maximum delay of one operation clock cycle.

The following figure shows when edge detection takes place.

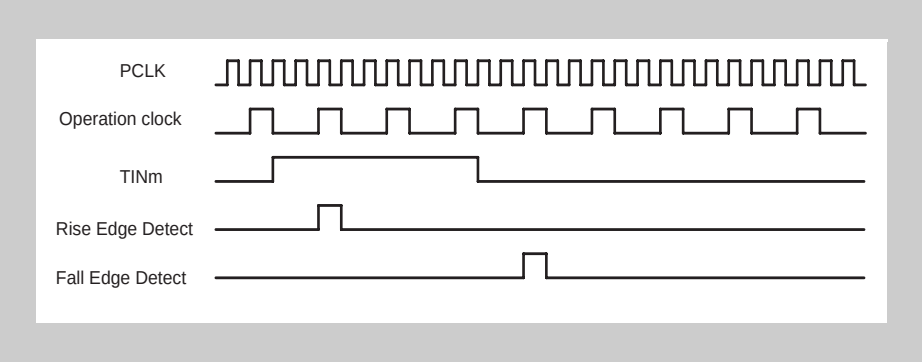


Figure 16-24 Basic edge detection timing

16.13 Independent Channel Operation Functions

The following sections list the independent channel operation functions provided by timer array unit B. For a general overview of independent channel operation, see *16.3 “Functional Description” on page 946*.

16.14 Independent Channel Interrupt Functions

This chapter describes functions that generate interrupts at regular intervals or with a specified delay.

- *16.14.1 “Interval timer function”*
- *16.14.2 “TAUBnTTINm Input interval timer function”*
- *16.14.3 “One-pulse output function”*

16.14.1 Interval timer function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUBnIm) at regular intervals. When an interrupt is generated, the TAUBnTTOUTm signal toggles, resulting in a square wave.

Prerequisites
- The operation mode must be set to interval timer mode, see *Table 16-12 "TAUBnCMORm settings for interval timer function" on page 982*.
- The channel output mode must be set to independent channel output mode 1, see *16.8 "Channel Output Modes" on page 962*.

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value.

When the counter reaches 0000_H, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles. TAUBnCNTm then reloads the TAUBnCDRm value and subsequently continues operation.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the counter starts to count down.

The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm stop but retain their values. The counter can be reset by setting TAUBnTS.TSm to 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm to 1 during operation.

Conditions If the TAUBnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUBnTTOUTm does not toggle. This results in an inverted TAUBnTTOUTm signal compared to when TAUBnCMORm.MD0 is set to 1. For details, see *16.10 "TAUBnTTOUTm Toggle and INTTAUBnIm Generation When Counter Start Is Triggered (MD0 Bit)" on page 971*.

(2) Equations

INTTAUBnIm cycle = count clock cycle x (TAUBnCDRm + 1)

TAUBnTTOUTm square wave cycle = count clock cycle x (TAUBnCDRm + 1) x 2

(3) Block diagram and general timing diagram

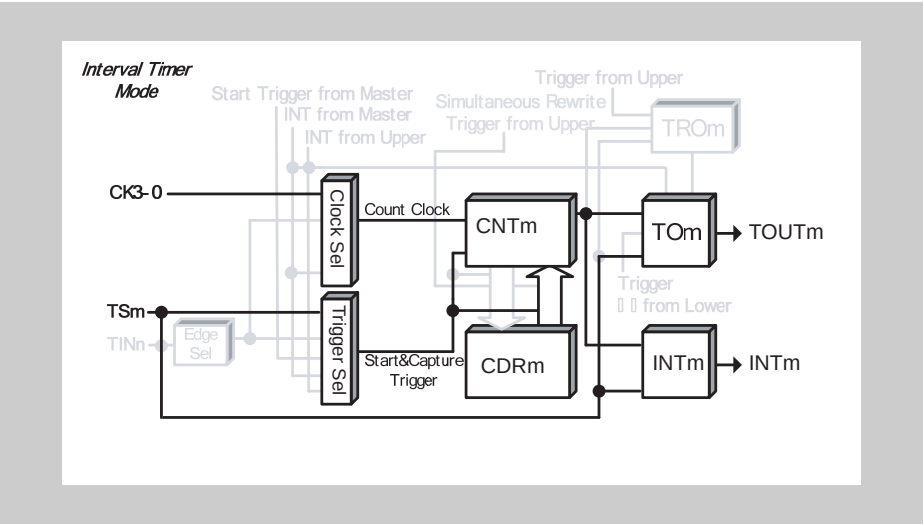


Figure 16-25 Block diagram for interval timer function

The following settings apply to the general timing diagram:

- INTTAUBnIm generated at operation start (TAUBnCMORm.MD0 = 1)

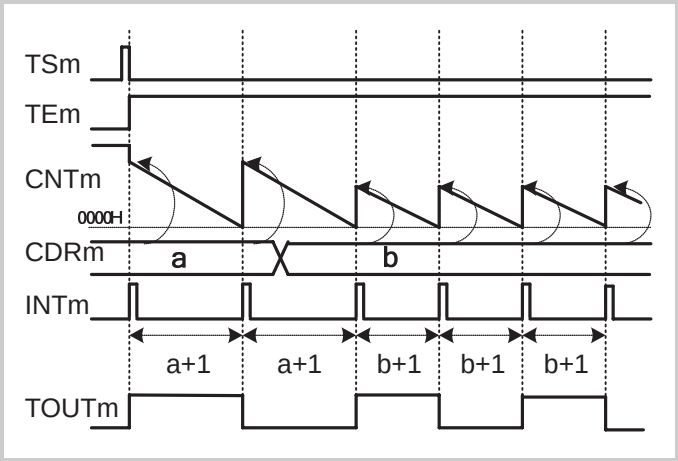


Figure 16-26 General timing diagram for interval timer function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-12 TAUBnCMORm settings for interval timer function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval timer mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start or restart 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start or restart

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-13 TAUBnCMURm settings for interval timer function

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Table 16-14 Control bit settings for independent channel output mode 1

Bit name	Setting
TOE.TOEm	1: Disables independent channel output mode controlled by software
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDL.TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

Note The channel output mode can also be set to channel output mode controlled by software by setting TAUBnTOE.TOEm = 0. TAUBnTTOUTm can then be controlled independently of the interrupts. For details, see *Table 16-10 "Channel output modes"*.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the interval timer function. Therefore, these registers must be set to 0.

Table 16-15 Simultaneous rewrite settings for interval timer function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for interval timer function

Table 16-16 Operating procedure for interval timer function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-12 "TAUBnCMORm settings for interval timer function" on page 982* and *Table 16-13 "TAUBnCMURm settings for interval timer function" on page 982*. Set the value of the TAUBnCDRm register. Set the channel output mode by setting the control bits as described in *Table 16-14 "Control bit settings for independent channel output mode 1" on page 983*.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value. When TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated and TAUBnTTOUTm toggles.
During operation	The TAUBnCDRm register value can be changed at any time. The TAUBnCNTm register can be read at all times.	TAUBnCNTm counts down. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated and TAUBnTTOUTm toggles.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

(a) TAUBnCDRm = 0000_H, count clock = PCLK/2

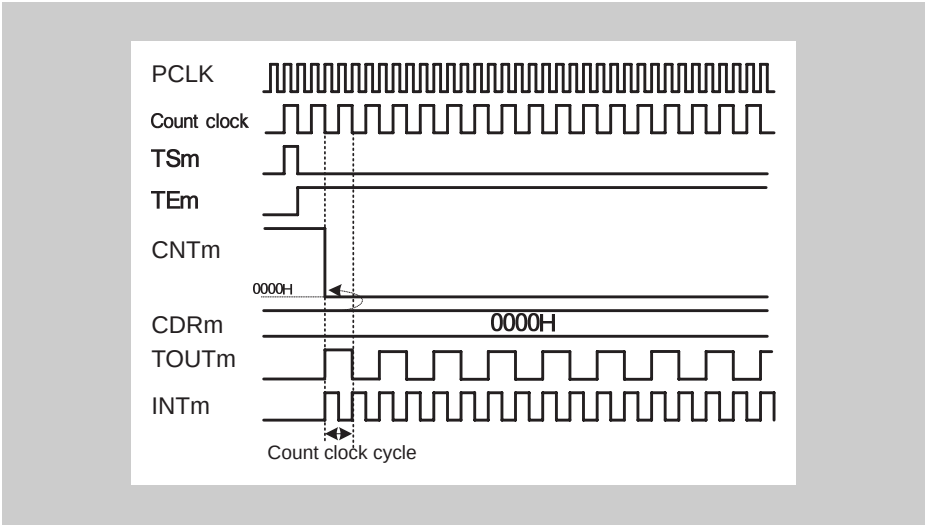


Figure 16-27 TAUBnCDRm = 0000_H, count clock = PCLK/2

- If TAUBnCDRm = 0000_H and the count clock = PCLK/2^1, the TAUBnCDRm value is written to TAUBnCNTm every count clock, meaning that TAUBnCNTm is always 0000_H.
- INTTAUBnIm is generated every count clock, resulting in TAUBnTTOUTm toggling every count clock.

(b) TAUBnCDRm = 0000_H, count clock = PCLK

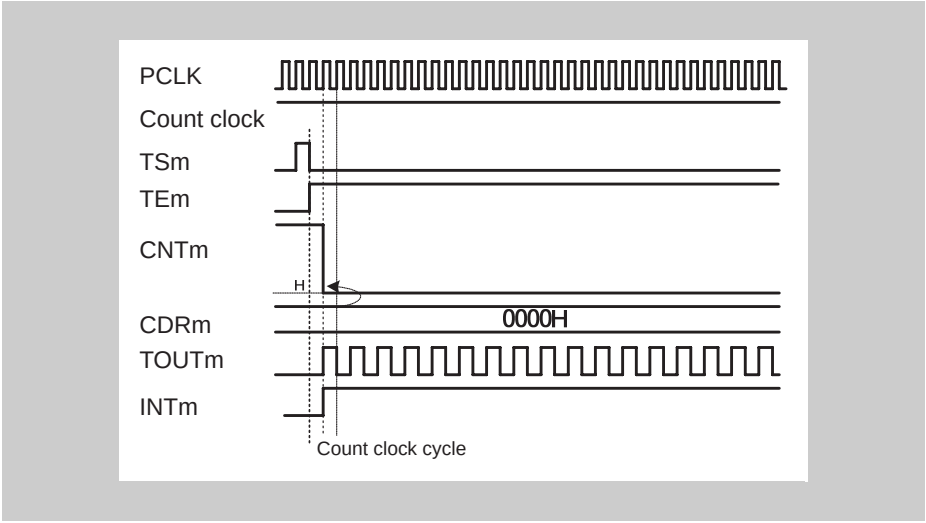


Figure 16-28 TAUBnCDRm = 0000_H, count clock = PCLK

- If TAUBnCDRm = 0000_H and the count clock = PCLK, the TAUBnCDRm value is written to TAUBnCNTm every PCLK clock, meaning that TAUBnCNTm is always 0000_H.
- INTTAUBnIm is generated continuously, resulting in TAUBnTTOUTm toggling every PCLK clock.

(c) Operation stop and restart

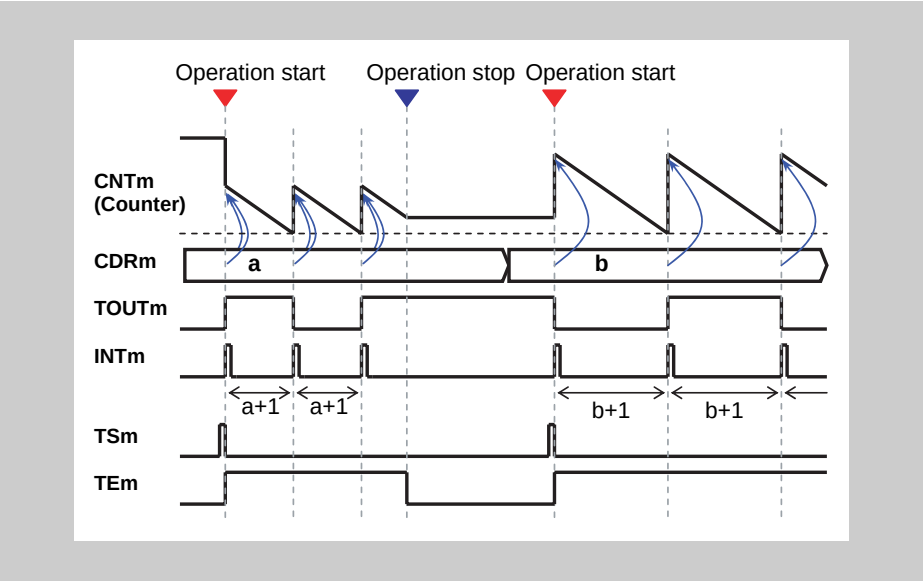


Figure 16-29 Operation stop and restart, TAUBnCMORm.MD0 = 1

- The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm and TAUBnTTOUTm stop but retain their values.
- The counter can be restarted by setting TAUBnTS.TSm to 1.

(d) Forced restart

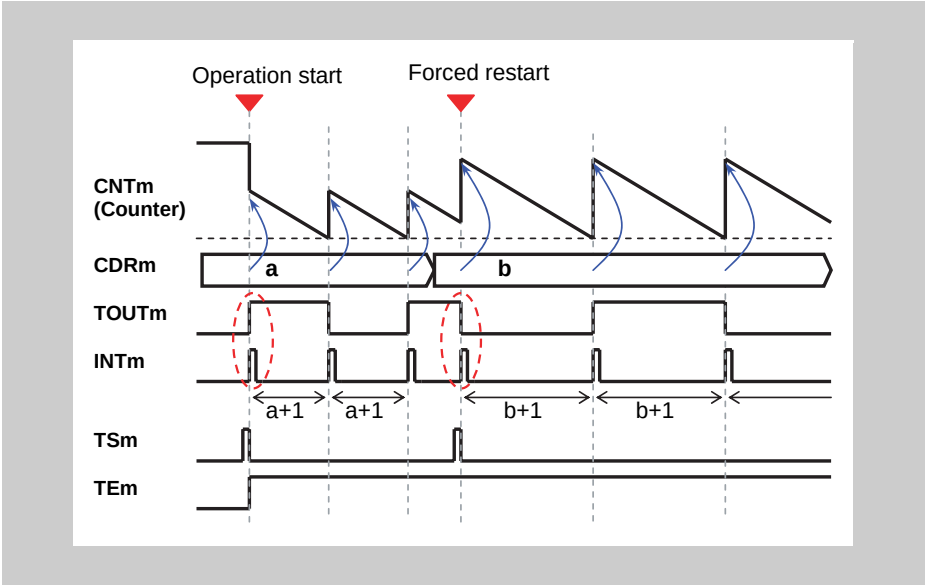


Figure 16-30 Forced restart operation, TAUBnCMORm.MD0 = 1

- The counter can be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm to 1 during operation.
- If the TAUBnCMORm.MD0 bit is set to 1, an interrupt at start or restart is generated and the output TOUTm toggles.

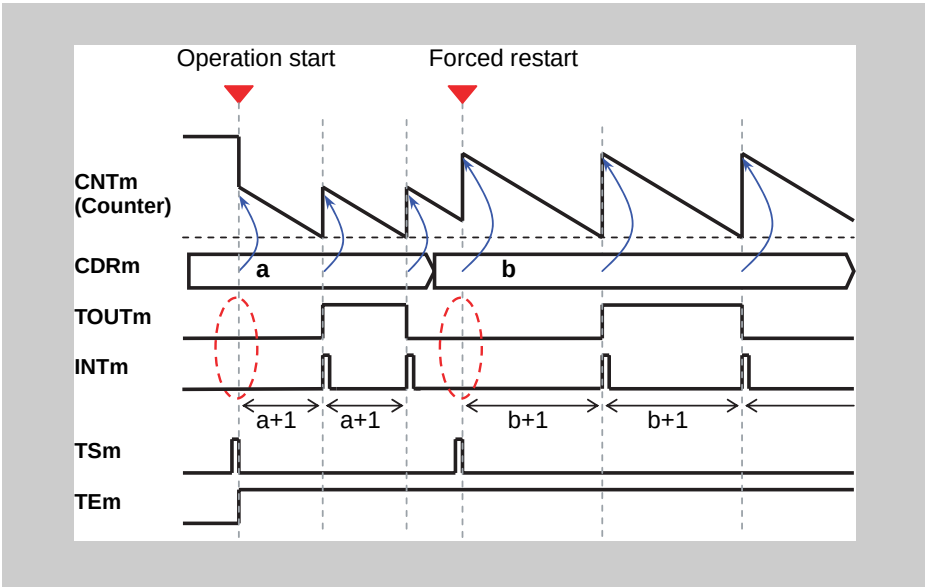


Figure 16-31 Forced restart operation, TAUBnCMORm.MD0 = 0

- The counter can be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm to 1 during operation.
- If the TAUBnCMORm.MD0 bit is set to 0, the interrupt at start or restart is not generated and TOUTm does not toggle.

16.14.2 TAUBnTTINm Input interval timer function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUBnIm) at regular intervals or when a valid TAUBnTTINm input edge is detected. When an interrupt is generated, the TAUBnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to interval timer mode, see *Table 16-17 "TAUBnCMORm settings for TAUBnTTINm input interval timer function" on page 990*.
- The channel output mode must be set to independent channel output mode 1, see *16.8 "Channel Output Modes" on page 962*.

Description This function operates in an identical manner to the interval timer function (see *16.14.1 "Interval timer function" on page 980*), except that this function is restarted by a valid TAUBnTTINm input edge. The type of edge used as the trigger is specified using the TAUBnCMURm.TIS[1:0] bits. Either rising edge, falling edge, or rising and falling edge can be selected.

(2) Equations

INTTAUBnIm cycle = count clock cycle x (TAUBnCDRm + 1)

TAUBnTTOUTm square wave cycle = count clock cycle x (TAUBnCDRm + 1) x 2

(3) Block diagram and general timing diagram

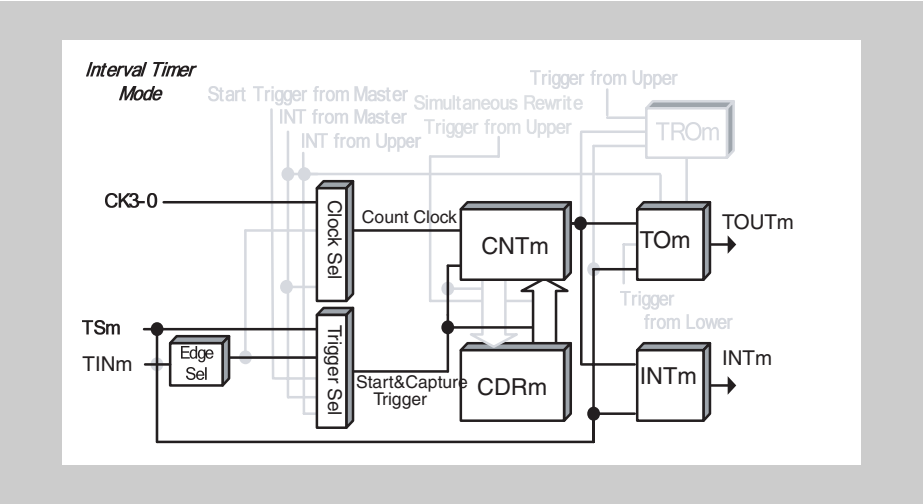


Figure 16-32 Block diagram for TAUBnTTINm Input interval timer function

The following settings apply to the general timing diagram:

- INTTAUBnIm generated at operation start (TAUBnCMORm.MD0 = 1) and valid edge detection of input signal.
- Rising edge detection (TAUBnCMURm.TIS[1:0] = 01_B)

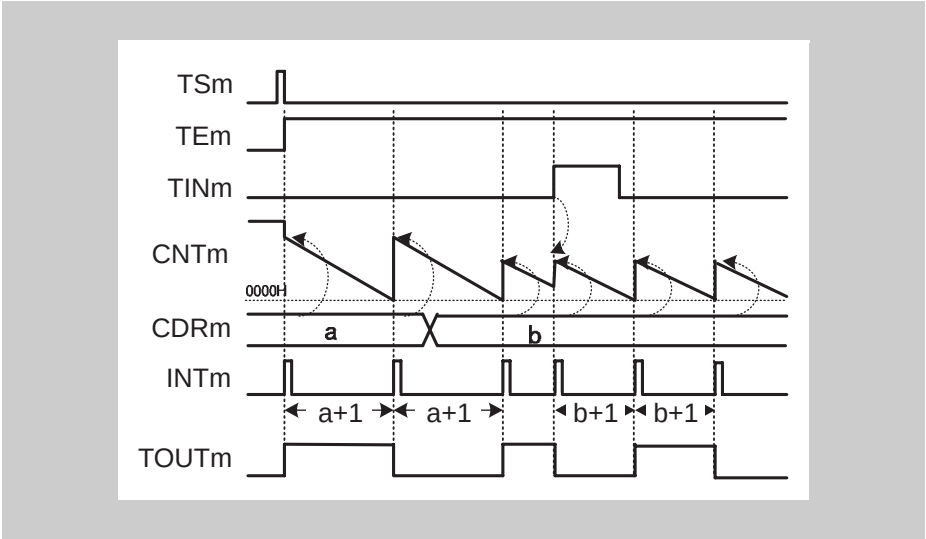


Figure 16-33 General timing diagram for TAUBnTTINm input interval timer function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-17 TAUBnCMORm settings for TAUBnTTINm input interval timer function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUBnTTINm input edge signal is used as the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval timer mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-18 TAUBnCMURm settings for TAUBnTTINm input interval timer function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Table 16-19 Control bit settings for independent channel output mode 1

Bit name	Setting
TOE.TOEm	1: Disables independent channel output mode controlled by software
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDL.TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

Note The channel output mode can also be set to channel output mode controlled by software by setting TAUBnTOE.TOEm = 0. TAUBnTTOUTm can then be controlled independently of the interrupts. For details, see *Table 16-10 "Channel output modes"*.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm Input interval timer function. Therefore, these registers must be set to 0.

Table 16-20 Simultaneous rewrite settings for TAUBnTTINm Input interval timer function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for TAUBnTTINm input interval timer function

Table 16-21 Operating procedure for TAUBnTTINm input interval timer function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-17 "TAUBnCMORm settings for TAUBnTTINm input interval timer function" on page 990* and *Table 16-18 "TAUBnCMURm settings for TAUBnTTINm input interval timer function" on page 990*. Set the value of the TAUBnCDRm register. Set the channel output mode by setting the control bits as described in *Table 16-19 "Control bit settings for independent channel output mode 1" on page 991*.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value. When TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated and TAUBnTTOUTm toggles.
During operation	The values of the TAUBnCMURm.TIS[1:0] and TAUBnTO.TOm bits and the TAUBnCDRm register can be changed at any time. The TAUBnCNTm register can be read at all times. Detection of TAUBnTTINm edge	TAUBnCNTm counts down. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated and TAUBnTTOUTm toggles When a TAUBnTTINm input valid edge is detected during count operation, TAUBnCNTm reloads the TAUBnCDRm value and continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

The timing diagrams in *16.14.1 "Interval timer function" on page 980* also apply, except for this function the counter can also be restarted by a valid TAUBnTTINm input edge.

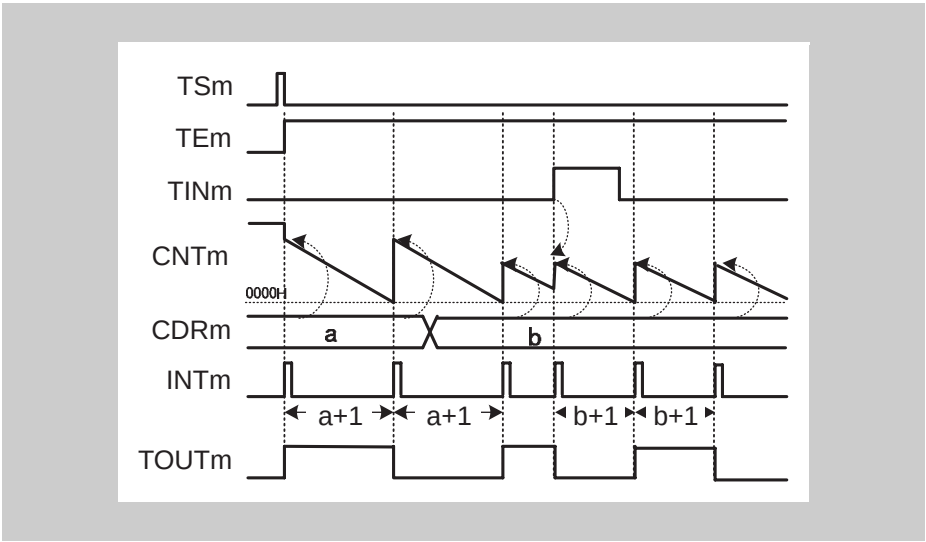


Figure 16-34 Counter triggered by rising TAUBnTTINm input edge (TAUBnCMURm.TIS[1:0] = 01_B), TAUBnCMORm.MD0 = 1

- If a valid TAUBnTTINm input edge is detected, an interrupt is generated which causes TAUBnTTOUTm to toggle. In this example, the valid edge is a rising edge (TAUBnCMURm.TIS[1:0] = 01_B).

16.14.3 One-pulse output function

(1) Overview

Summary This function generates an interrupt (INTTAUBnIm) when a valid TAUBnTTINm input edge is detected and also a specific interval later. TAUBnTTINm input signal pulses that occur within the defined interval are ignored. When an interrupt is generated, the TAUBnTTOUTm signal toggles, resulting in a square wave.

Prerequisites
- The operation mode must be set to pulse one-count mode, see *Table 16-22 "TAUBnCMORm settings for one-pulse output function" on page 996*
- The channel output mode must be set to independent channel output mode 1, see *16.8 "Channel Output Modes" on page 962*.
- Trigger detection must be disabled during counting (TAUBnCMORn.MD0 = 0).

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUBnTTINm input edge is detected. The value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from the TAUBnCDRm value. An interrupt is generated and TAUBnTTOUTm toggles.

When the counter reaches 0001_H an interrupt is generated and TAUBnTTOUTm toggles. The counter stops at 0000_H and awaits the next valid TAUBnTTINm input edge.

When the counter is counting down, further TAUBnTTINm input signals are ignored, i.e. the counter does not reset.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUBnCMURm.TIS[1:0] bits:
- If TAUBnCMURm.TIS[1:0] = 00_B, falling edges trigger the counter.
- If TAUBnCMURm.TIS[1:0] = 01_B, rising edges trigger the counter.
- If TAUBnCMURm.TIS[1:0] = 10_B, rising and falling edges trigger the counter.

(2) Equations

Interval between TAUBnTTINm and INTTAUBnIm = TAUBnTTOUTm (timer output) width = count clock cycle × TAUBnCDRm

(3) Block diagram and general timing diagram

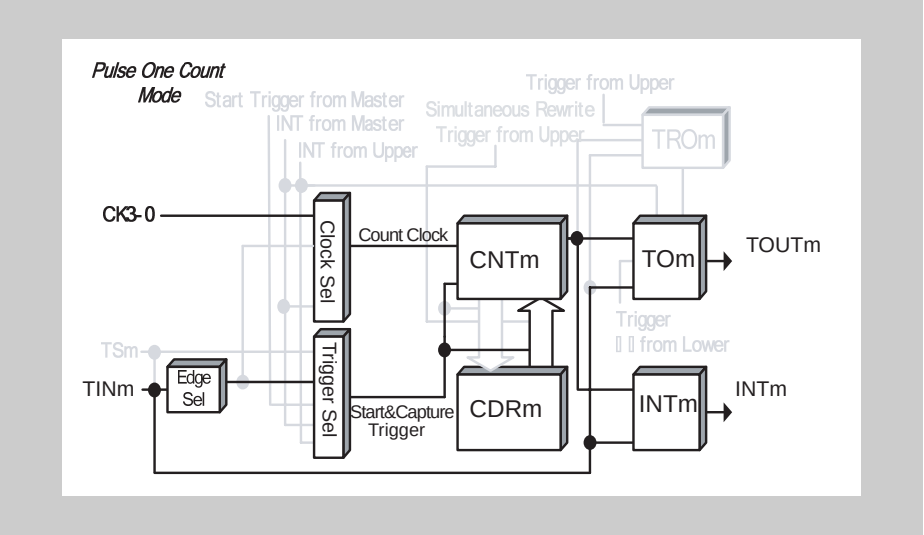


Figure 16-35 Block diagram for one-pulse output function

The following settings apply to the general timing diagram:

- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

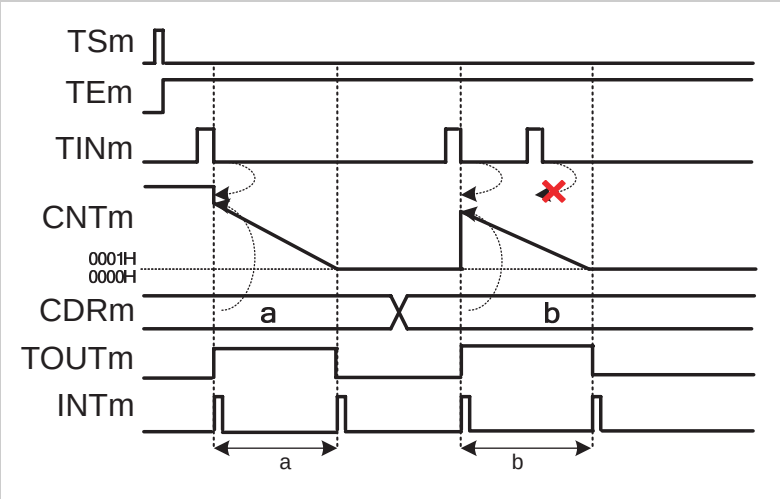


Figure 16-36 General timing diagram for one-pulse output function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-22 TAUBnCMORm settings for one-pulse output function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUBnTTINm input edge signal is used as the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1010: Pulse one-count mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-23 TAUBnCMURm settings for one-pulse output function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode

Table 16-24 Control bit settings for independent channel output mode 1

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	0: Independent channel output
TOCm	1: Independent channel output mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

Note The channel output mode can also be set to channel output mode controlled by software by setting TAUBnTOE.TOEm = 0. TAUBnTTOUTm can then be controlled independently of the interrupts. For details, see *Table 16-10 “Channel output modes” on page 963*.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the one-pulse output function. Therefore, these registers must be set to 0.

Table 16-25 Simultaneous rewrite settings for one-pulse output function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for one-pulse output function

Table 16-26 Operating procedure for one-pulse output function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-22 "TAUBnCMORm settings for one-pulse output function" on page 996* and *Table 16-23 "TAUBnCMURm settings for one-pulse output function" on page 996*. Set the value of the TAUBnCDRm register. Set the channel output mode by setting the control bits as described in *Table 16-24 "Control bit settings for independent channel output mode 1" on page 997*.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. When a start edge is detected, TAUBnCNTm loads the TAUBnCDRm value.
During operation	The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at all times.	INTTAUBnIm is generated when TAUBnCNTm starts and TAUBnTTOUTm is set to its active level. TAUBnCNTm counts down. When the counter reaches 0001_H: • INTTAUBnIm is generated • TAUBnTTOUTm is set to its inactive level. TAUBnCNTm stops counting and waits for a trigger. If a trigger occurs while TAUBnCNTm is counting, the trigger is ignored. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values.

Restart

16.15 Independent Channel Signal Measurement Functions

This chapter describes functions that measure the widths of an individual TAUBnTTINm pulse or the total width of successive TAUBnTTINm pulses. It also describes functions that measure the interval of the signal or that compare the width of a pulse with a reference value.

- *16.15.1 “TAUBnTTINm input pulse interval measurement function”*
- *16.15.2 “TAUBnTTINm input signal width measurement function”*
- *16.15.3 “Overflow interrupt output function (during TAUBnTTINm width measurement)”*
- *16.15.4 “TAUBnTTINm input period count detection function”*
- *16.15.5 “Overflow interrupt output function (during TAUBnTTINm input period count detection)”*
- *16.15.6 “TAUBnTTINm input pulse interval judgment function”*
- *16.15.7 “TAUBnTTINm input signal width judgment function”*

16.15.1 TAUBnTTINm input pulse interval measurement function

(1) Overview

Summary This function captures the count value and uses this value and the overflow bit TAUBnCSRm.OVF to measure the interval of the TAUBnTTINm input signal.

Prerequisites

- The operation mode must be set to capture mode, see *Table 16-28 "TAUBnCMORm settings for TAUBnTTINm input pulse interval measurement function" on page 1002*.
- TAUBnTTOUTm is not used for this function.

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The counter TAUBnCNTm starts counting up from 0000_H. When a valid TAUBnTTINm edge is detected, the value of TAUBnCNTm is captured, transferred to TAUBnCDRm, and an interrupt INTTAUBnIm is generated. The counter resets to 0000_H and subsequently continues operation.

If the counter reaches $FFFF_H$ before a valid TAUBnTTINm edge is detected, it overflows to 0000_H. The counter is reset to 0000_H and subsequently continues operation. The values transferred to TAUBnCDRm and TAUBnCSRm.OVF respectively depend on the values of bits TAUBnCMORm.COS[1:0]:

Table 16-27 Effects of an overflow

TAUBnCMORm.COS[1:0]	When overflow occurs		When a valid TAUBnTTINm input is then detected	
	TAUBnCDRm	TAUBnCSRm.OVF	TAUBnCDRm and TAUBnCNTm	TAUBnCSRm.OVF
00	Unchanged	0	TAUBnCNTm written to TAUBnCDRm	1
01		1		
10	Set to $FFFF_H$	0	TAUBnCNTm set to 0, TAUBnCDRm unchanged	0
11		1		

If an overflow is set (TAUBnCSRm.OVF = 1), it can only be cleared by a CPU command that sets TAUBnCSCm.CLOV = 1.

The combination of the value of TAUBnCDRm and TAUBnCSRm.OVF can be used to deduce the interval of the TAUBnTTINm signal. However, if an overflow occurs multiple times before a valid TAUBnTTINm input is detected, the overflow bit TAUBnCSRm.OVF cannot indicate this.

The function can be stopped by setting TAUBnTT.TTm = 1, which in turn sets TAUBnTE.TEm = 0. TAUBnCNTm stops but retains its value. While the function is stopped, TAUBnTTINm input valid edge detection and TAUBnCNTm capture are not performed.

The function can be restarted by setting TAUBnTS.TSm = 1. The counter is reset to 0000_H and subsequently continues operation. The counter can also be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm = 1 during operation.

Conditions If the TAUBnCMORm.MD0 bit is set to 0, the interrupt at start or restart is not generated. For details, see *16.10 "TAUBnTTOUTm Toggle and INTTAUBnIm Generation When Counter Start Is Triggered (MD0 Bit)" on page 971*.

Note When TAUBnCMORm.COS[1:0] = 11_B, the value of TAUBnCNTm is *not* written to TAUBnCDRm when the first valid TAUBnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUBnTTINm input pulse interval = count clock cycle x
[(TAUBnCSRm.OVF x ($FFFF_H$ + 1)) + TAUBnCDRm capture value + 1]

(3) Block diagram and general timing diagram

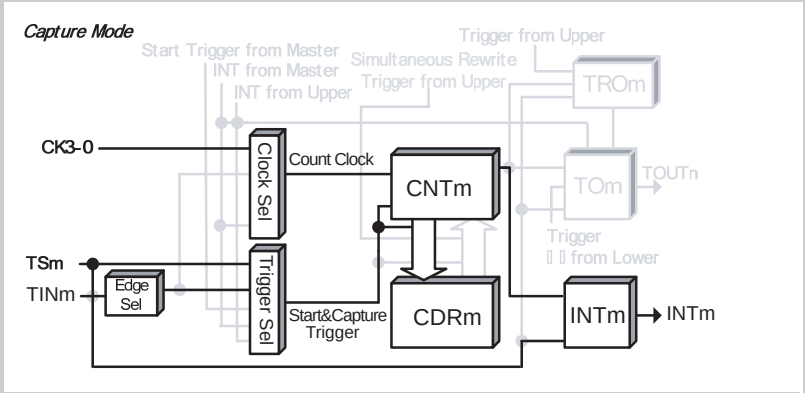


Figure 16-37 Block diagram for TAUBnTTINm input pulse interval measurement function

The following settings apply to the general timing diagram:

- INTTAUBnIm not generated at operation start (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)
- When a valid TAUBnTTINm input is detected after an overflow TAUBnCDRm is changed and TAUBnCSRm.OVF is set to 1 (TAUBnCMORm.COS[1:0] = 00_B)

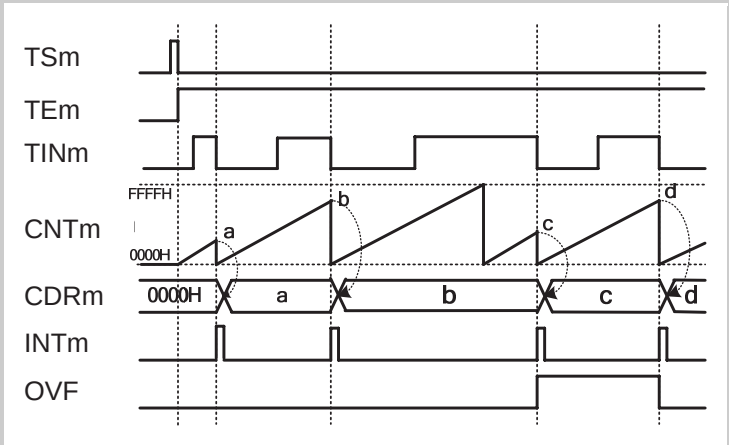


Figure 16-38 General timing diagram for TAUBnTTINm input pulse interval measurement function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-28 TAUBnCMORm settings for TAUBnTTINm input pulse interval measurement function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid edge of the TAUBnTTINm input signal is the external capture trigger
COS[1:0]	See *Table 16-27 "Effects of an overflow" on page 1000*.
MD[4:1]	0010: Capture mode
MD0	0: INTTAUBnIm not generated at operation start 1: Generates INTTAUBnIm at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-29 TAUBnCMURm settings for TAUBnTTINm input pulse interval measurement function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simuBtaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm input pulse interval measurement function. Therefore, these registers must be set to 0.

Table 16-30 Simultaneous rewrite settings for TAUBnTTINm input pulse interval measurement function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for TAUBnTTINm input pulse interval measurement function

Table 16-31 Operating procedure for TAUBnTTINm input pulse interval measurement function

		Operation	Status of TAUBn
	Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-28 "TAUBnCMORm settings for TAUBnTTINm input pulse interval measurement function" on page 1002* and *Table 16-29 "TAUBnCMURm settings for TAUBnTTINm input pulse interval measurement function" on page 1002*. Set the value of the TAUBnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Restart	Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm is cleared to 0000_H. INTTAUBnIm is generated when TAUBnCMORm.MD0 is set to 1.
	During operation	Detection of TAUBnTTINm edges. The TAUBnCMURm.TIS[1:0] bits can be changed at any time. The TAUBnCDRm and TAUBnCSRm registers can be read at any time.	TAUBnCNTm starts to count up from 0000_H. When a TAUBnTTINm valid edge is detected: • TAUBnCNTm transfers (captures) its value to TAUBnCDRm, and returns to 0000_H • INTTAUBnIm is then generated. Afterwards, this procedure is repeated.
	Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnCSRm.OVF retain their current values.

(6) Specific timing diagrams: overflow behavior

(a) TAUBnCMORm.COS[1:0] = 00_B

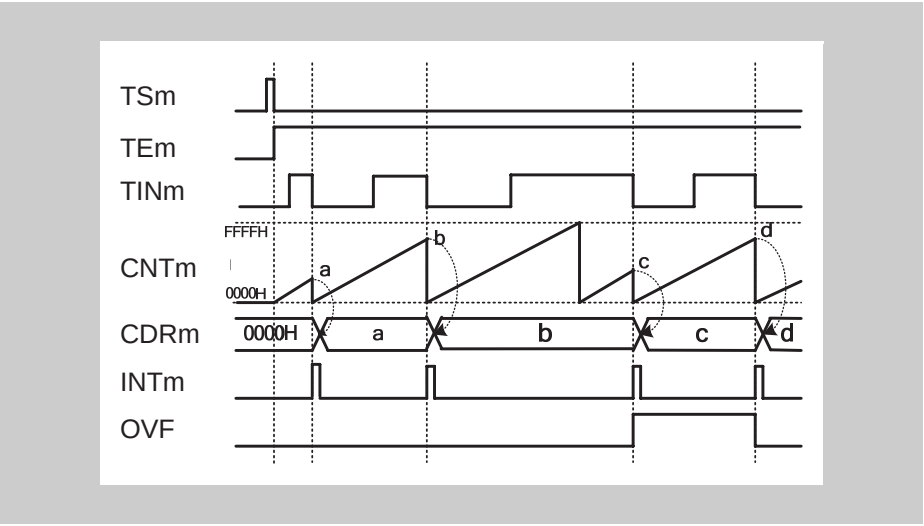


Figure 16-39 TAUBnCMORm.COS[1:0] = 00_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=00_B

- When an overflow occurs, the value of TAUBnCDRm remains unchanged and TAUBnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUBnTTINm input edge, the value of TAUBnCNTm is written to TAUBnCDRm and TAUBnCSRm.OVF is set to 1.

(b) TAUBnCMORm.COS[1:0] = 01_B

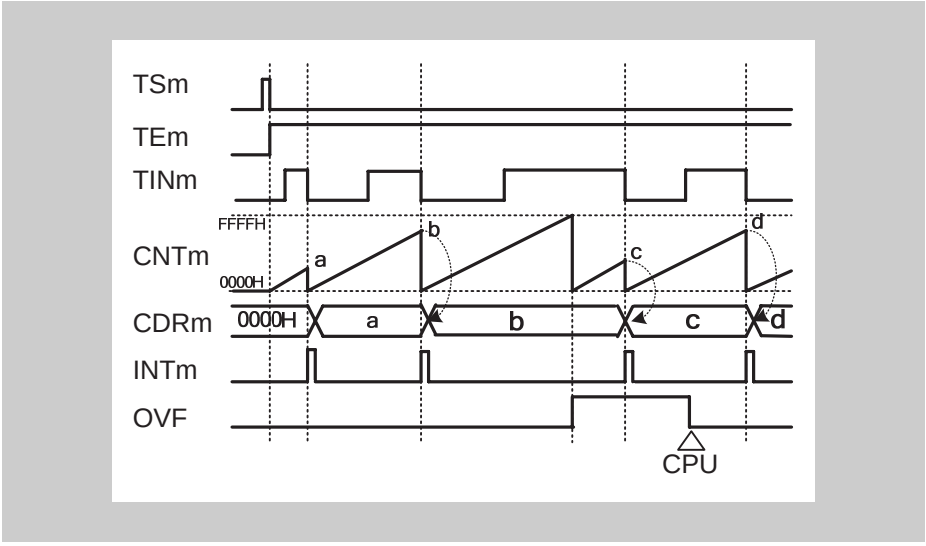


Figure 16-40 TAUBnCMORm.COS[1:0] = 01_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=00_B

- When an overflow occurs, the value of TAUBnCDRm remains unchanged and TAUBnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUBnTTINm input edge, the value of TAUBnCNTm is written to TAUBnCDRm.
- TAUBnCSRm.OVF is only cleared by a CPU command.

(c) TAUBnCMORm.COS[1:0] = 10_B

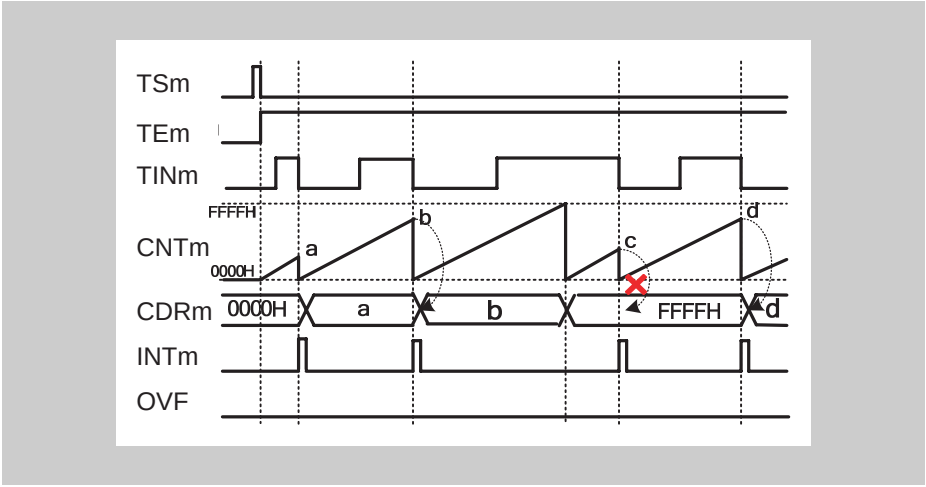


Figure 16-41 TAUBnCMORm.COS[1:0] = 10_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=00_B

- When an overflow occurs, TAUBnCDRm is set to $FFFF_H$ and TAUBnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUBnTTINm input edge, TAUBnCNTm is reset to 0, but TAUBnCDRm and TAUBnCSRm.OVF remain unchanged.
- Thus, the next TAUBnTTINm input valid edge after the overflow is ignored.

(d) TAUBnCMORm.COS[1:0] = 11_B

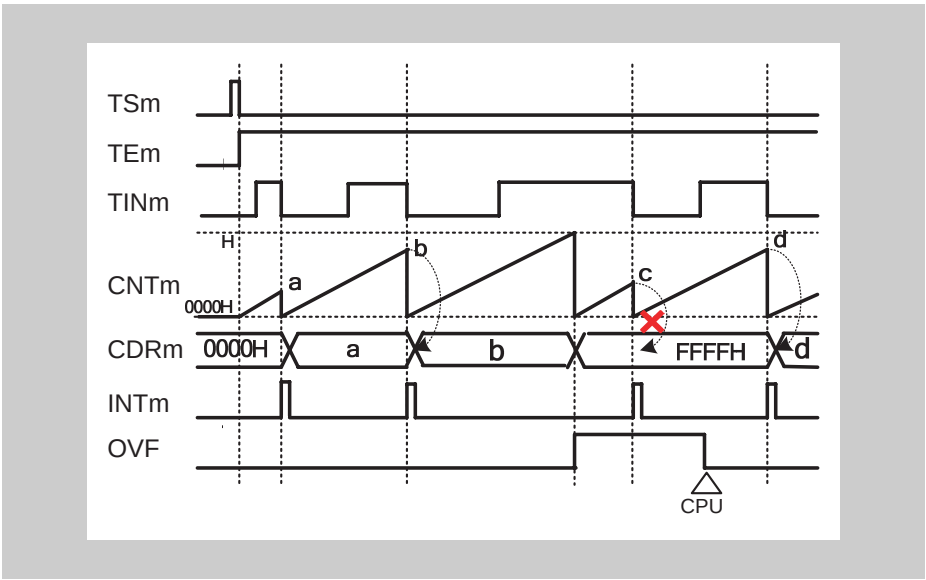


Figure 16-42 TAUBnCMORm.COS[1:0] = 11_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=00_B

- When an overflow occurs, TAUBnCDRm is set to $FFFF_H$, and TAUBnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUBnTTINm input edge, TAUBnCNTm is reset to 0, but TAUBnCDRm and TAUBnCSRm.OVF remain unchanged.
- Thus, the next TAUBnTTINm input valid edge after the overflow is ignored.
- TAUBnCSRm.OVF is cleared by a CPU command.

16.15.2 TAUBnTTINm input signal width measurement function

(1) Overview

Summary This function measures the width of a TAUBnTTINm input signal.

Prerequisites

- The operation mode must be set to capture & one-count mode, see *Table 16-33 "TAUBnCMORm settings for TAUBnTTINm input signal width measurement function" on page 1010*.
- TAUBnTTOUTm is not used for this function.
- TAUBnCMORm.MD0 must be set to 0.

Description

<R> The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. When a valid TAUBnTTINm start edge is detected, the counter TAUBnCNTm starts counting up from 0000_H. When a valid TAUBnTTINm stop edge is detected, the value of TAUBnCNTm is captured, transferred to TAUBnCDRm, and an interrupt INTTAUBnIm is generated. The counter retains its value (TAUBnCDRm + 1) and awaits the next valid TAUBnTTINm input start edge.

If the counter reaches $FFFF_H$ before a valid TAUBnTTINm stop edge is detected, it overflows. The counter is reset to 0000_H and subsequently continues operation. The values transferred to TAUBnCDRm and TAUBnCSRm.OVF respectively depend on the values of bits TAUBnCMORm.COS[1:0]:

Table 16-32 Effects of an overflow

TAUBnCMORm.COS[1:0]	When overflow occurs		When a valid TAUBnTTINm input stop edge is detected	
	TAUBnCDRm	TAUBnCSRm.OVF	TAUBnCDRm and TAUBnCNTm	TAUBnCSRm.OVF
00	Unchanged	0	TAUBnCNTm written to TAUBnCDRm	1
01		1		
10	Set to $FFFF_H$	0	TAUBnCNTm stops counting TAUBnCDRm unchanged	0
11		1		

If an overflow is set (TAUBnCSRm.OVF = 1), it can only be cleared by a CPU command that sets TAUBnCSCm.CLOV = 1.

The combination of the value of TAUBnCDRm and TAUBnCSRm.OVF can be used to deduce the width of the TAUBnTTINm signal. However, if an overflow occurs multiple times before a valid TAUBnTTINm input is detected, the overflow bit TAUBnCSRm.OVF cannot indicate this.

This function cannot be forcibly restarted.

Note When TAUBnCMORm.COS[1:0] = 11_B, the value of TAUBnCNTm is *not* written to TAUBnCDRm when the first valid TAUBnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUBnTTINm input signal width = count clock cycle x [(TAUBnCSRm.OVF x ($FFFF_H$ + 1)) + TAUBnCDRm capture value + 1]

(3) Block diagram and general timing diagram

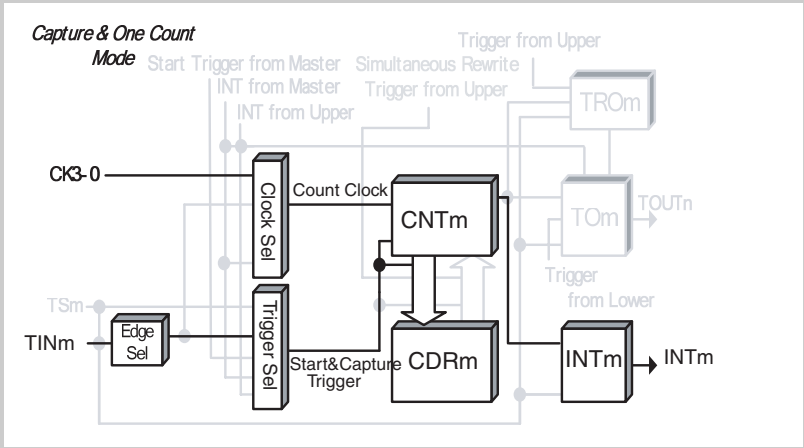


Figure 16-43 Block diagram for TAUBnTTINm input signal width measurement function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUBnCMURm.TIS[1:0] = 11_B)
- When a valid TAUBnTTINm input is detected after an overflow TAUBnCDRm is changed and TAUBnCSRm.OVF is set to 1 (TAUBnCMORm.COS[1:0] = 00_B)

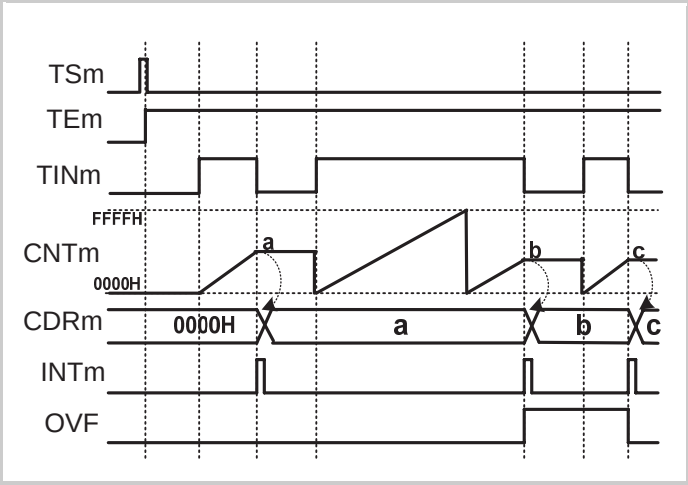


Figure 16-44 General timing diagram for TAUBnTTINm input signal width measurement function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-33 TAUBnCMORm settings for TAUBnTTINm input signal width measurement function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	See *Table 16-32 “Effects of an overflow” on page 1008*
MD[4:1]	0110: Capture & one-count mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-34 TAUBnCMURm settings for TAUBnTTINm input signal width measurement function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm input signal width measurement function. Therefore, these registers must be set to 0.

Table 16-35 Simultaneous rewrite settings for TAUBnTTINm input signal width measurement function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for TAUBnTTINm input signal width measurement function

Table 16-36 Operating procedure for TAUBnTTINm input signal width measurement function

<R>

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-33 "TAUBnCMORm settings for TAUBnTTINm input signal width measurement function" on page 1010* and *Table 16-34 "TAUBnCMURm settings for TAUBnTTINm input signal width measurement function" on page 1010*. Set the value of the TAUBnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. When a TAUBnTTINm start is detected, TAUBnCNTm starts to count up.
During operation	Detection of TAUBnTTINm edges. The TAUBnCMURm.TIS[1:0] bits can be changed at any time. The TAUBnCDRm and TAUBnCSRm registers can be read at any time.	TAUBnCNTm starts to count up from 0000_H. When a TAUBnTTINm valid edge is detected: • TAUBnCNTm transfers (captures) its value to TAUBnCDRm, and INTTAUBnIm is then generated • Counting stops at the "value transferred to TAUBnCDRm + 1" value and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUBnCMORm.COS[1:0] = 00_B

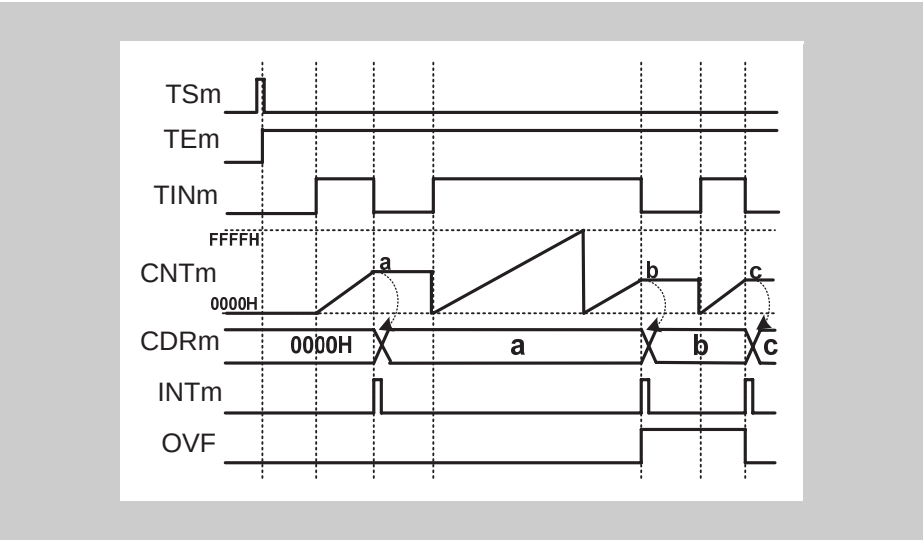


Figure 16-45 TAUBnCMORm.COS[1:0] = 00_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=11_B

- When an overflow occurs, the value of TAUBnCDRm remains unchanged and TAUBnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUBnTTINm input edge, the value of TAUBnCNTm is written to TAUBnCDRm and TAUBnCSRm.OVF is set to 1.

(b) TAUBnCMORm.COS[1:0] = 01_B

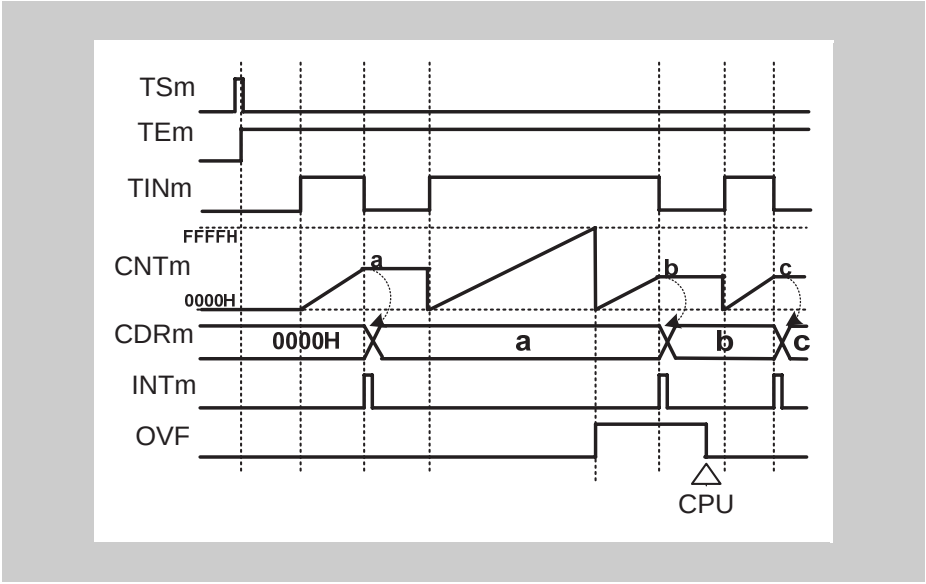


Figure 16-46 TAUBnCMORm.COS[1:0] = 01_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=11_B

- When an overflow occurs, the value of TAUBnCDRm remains unchanged and TAUBnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUBnTTINm input edge, the value of TAUBnCNTm is written to TAUBnCDRm.
- TAUBnCSRm.OVF is only cleared by a CPU command.

(c) TAUBnCMORm.COS[1:0] = 10_B

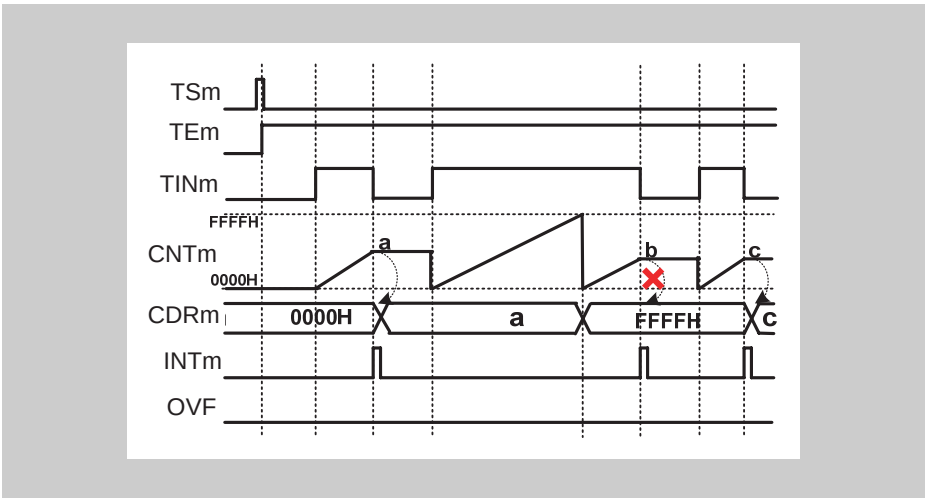


Figure 16-47 TAUBnCMORm.COS[1:0] = 10_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=11_B

- When an overflow occurs, TAUBnCDRm is set to $FFFF_H$ and TAUBnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUBnTTINm input edge, TAUBnCNTm is reset to 0, but TAUBnCDRm and TAUBnCSRm.OVF remain unchanged.
- Thus, the next TAUBnTTINm input valid edge after the overflow is ignored.

(d) TAUBnCMORm.COS[1:0] = 11_B

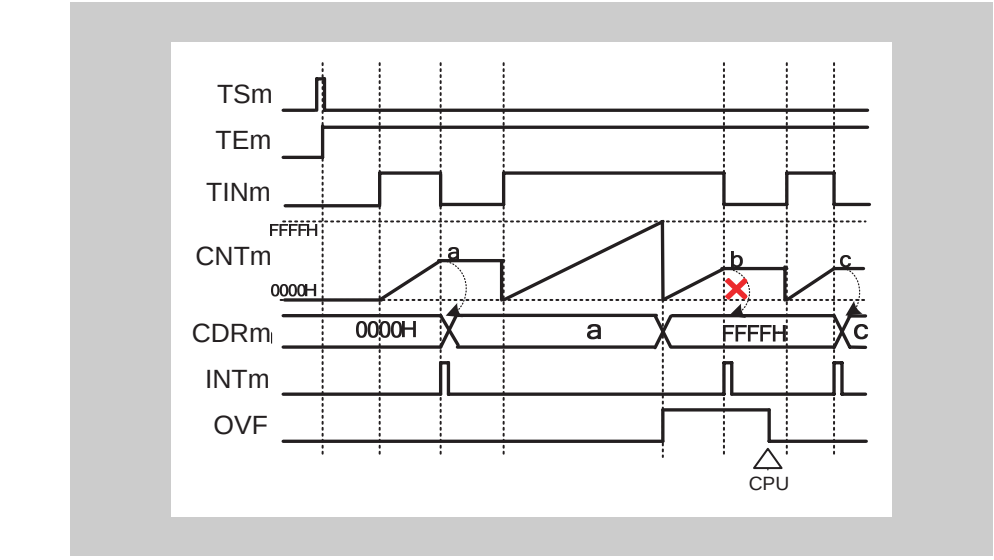


Figure 16-48 TAUBnCMORm.COS[1:0] = 11_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=11_B

- When an overflow occurs, TAUBnCDRm is set to $FFFF_H$, and TAUBnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUBnTTINm input edge, TAUBnCNTm is reset to 0, but TAUBnCDRm and TAUBnCSRm.OVF remain unchanged.
- Thus, the next TAUBnTTINm input valid edge after the overflow is ignored.
- TAUBnCSRm.OVF is cleared by a CPU command.

16.15.3 Overflow interrupt output function (during TAUBnTTINm width measurement)

(1) Overview

Summary This function measures the width of an individual TAUBnTTINm input signal. An interrupt is generated if the TAUBnTTINm input width is longer than $FFFF_H$.

Prerequisites
- The operation mode must be set to one-count mode, see *Table 16-37 "TAUBnCMORm settings for overflow interrupt output function (during TAUBnTTINm width measurement)" on page 1018*.
- TAUBnTTOUTm is not used for this function.
- The value of TAUBnCDRm must be set to $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUBnTTINm input start edge is detected. $FFFF_H$ is written to TAUBnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value.

When the next TAUBnTTINm input start edge is detected, TAUBnCNTm reloads $FFFF_H$ and starts to count down.

If the counter reaches 0000_H before a stop edge is detected, an interrupt is generated.

Conditions The valid start and stop edges are specified by the TAUBnCMURm.TIS[1:0] bits:
- If TAUBnCMURm.TIS[1:0] = 10_B, the TAUBnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUBnCMURm.TIS[1:0] = 11_B, the TAUBnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

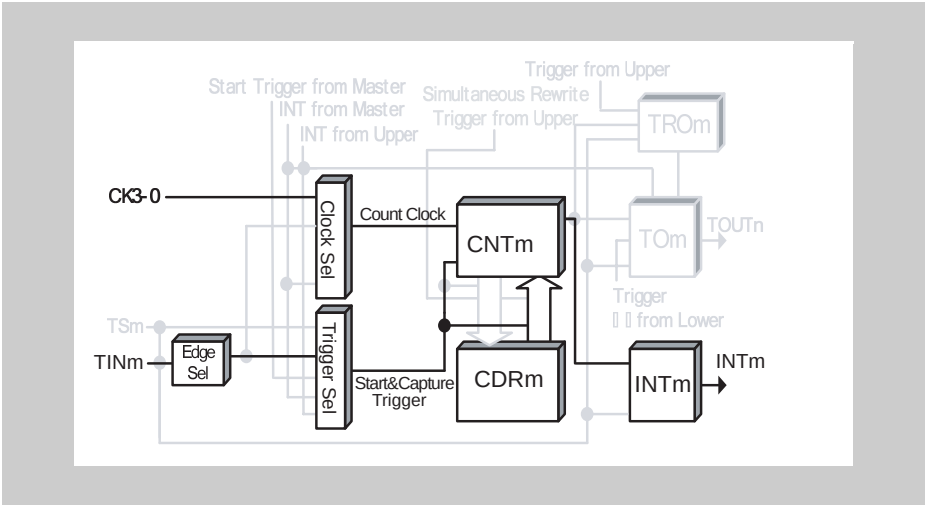


Figure 16-49 Block diagram for overflow interrupt output function (during TAUBnTTINm width measurement)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUBnCMURm.TIS[1:0] = 11_B)

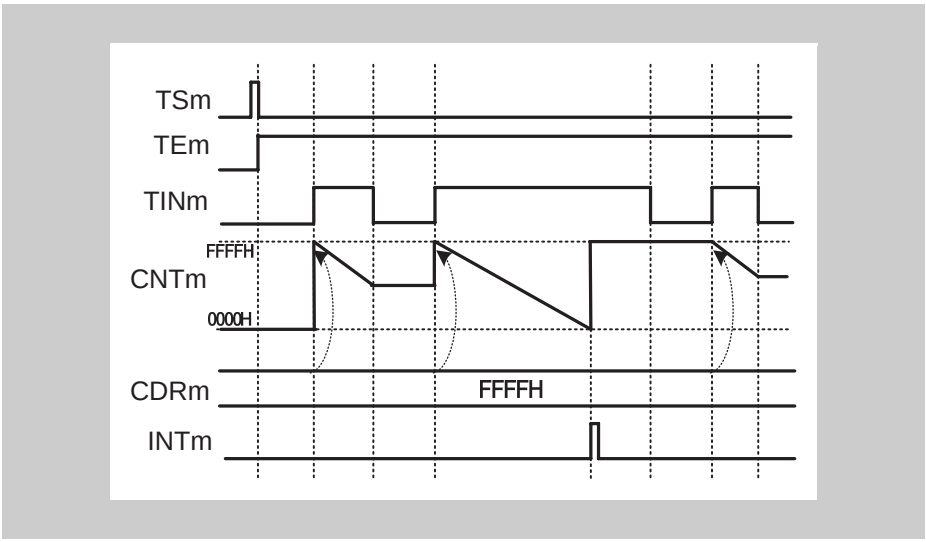


Figure 16-50 General timing diagram for overflow interrupt output function (during TAUBnTTINm width measurement)

(3) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-37 TAUBnCMORm settings for overflow interrupt output function (during TAUBnTTINm width measurement)

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One-count mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-38 TAUBnCMURm settings overflow interrupt output function (during TAUBnTTINm width measurement)

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the overflow interrupt output function (during TAUBnTTINm width measurement). Therefore, these registers must be set to 0.

(4) Operating procedure for overflow interrupt output function (during TAUBnTTINm width measurement)

Table 16-39 Simultaneous rewrite settings for overflow interrupt output function (during TAUBnTTINm width measurement)

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

Table 16-40 Operating procedure for overflow interrupt output function (during TAUBnTTINm width measurement)

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-37 "TAUBnCMORm settings for overflow interrupt output function (during TAUBnTTINm width measurement)" on page 1018* and *Table 16-38 "TAUBnCMURm settings overflow interrupt output function (during TAUBnTTINm width measurement)" on page 1018*. Set the value of the TAUBnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the start edge. When a start edge is detected, TAUBnCNTm loads the TAUBnCDRm value ($FFFF_H$).
During operation	The TAUBnCNTm register can be read at any time. Detection of TAUBnTTINm edges.	TAUBnCNTm counts down. When the counter reaches 0000_H: • INTTAUBnIm is generated • TAUBnCNTm stops counting at $FFFF_H$ and waits for a trigger. When a reverse edge of TAUBnTTINm is detected during count operation: • TAUBnCNTm stops counting and waits for a trigger. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

16.15.4 TAUBnTTINm input period count detection function

(1) Overview

Summary This function measures the cumulative width of a TAUBnTTINm input signal.

Prerequisites
- The operation mode must be set to capture & gate count mode, see *Table 16-41 "TAUBnCMORm settings for TAUBnTTINm input period count detection function" on page 1023*.
- TAUBnTTOUTm is not used for this function

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The counter awaits a valid TAUBnTTINm input edge.

When a valid TAUBnTTINm input start edge is detected, the counter starts to count from 0000_H.

When a valid TAUBnTTINm input stop edge is detected, the current TAUBnCNTm value is written to TAUBnCDRm and an interrupt (INTTAUBnIm) is generated. The counter stops and retains its value (TAUBnCDRm + 1) until the next valid TAUBnTTINm input start edge is detected.

If the counter reaches $FFFF_H$ the bit TAUBnCSRm.OVF is set to 1 and the counter restarts from 0000_H. The value of TAUBnCSRm.OVF is reset by the CPU by setting TAUBnCSCm.CLOV = 1.

Note The input TAUBnTTINm is sampled at the frequency of the operation clock, specified by the TAUBnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUBnTTOUTm has an error of ± 1 operation clock cycle.

Conditions The valid start and stop edges are specified by the TAUBnCMURm.TIS[1:0] bits:
- If TAUBnCMURm.TIS[1:0] = 10_B, the TAUBnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUBnCMURm.TIS[1:0] = 11_B, the TAUBnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

(2) Equations

Cumulative TAUBnTTINm input width =
count clock cycle × [(($FFFF_H$ + 1) × TAUBnCSRm.OVF) + (TAUBnCDRm capture value + 1)]

(3) Block diagram and general timing diagram

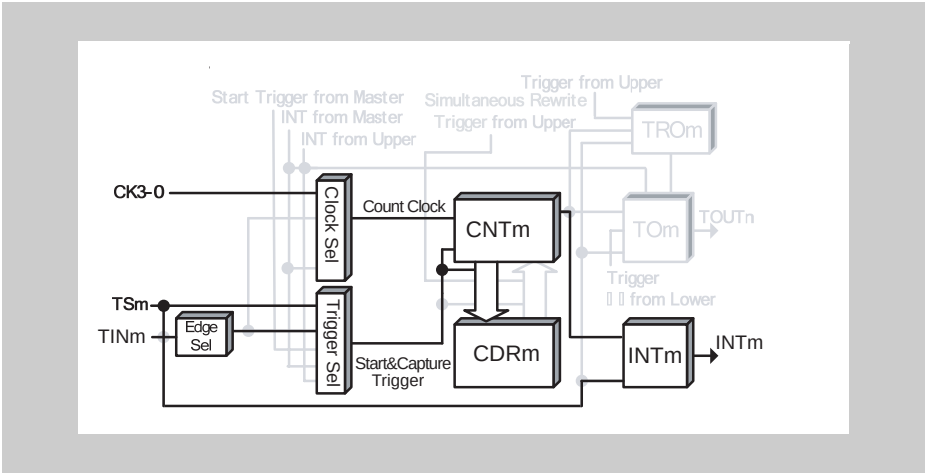


Figure 16-51 Block diagram for TAUBnTTINm input period count detection function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUBnCMURm.TIS[1:0] = 11_B)

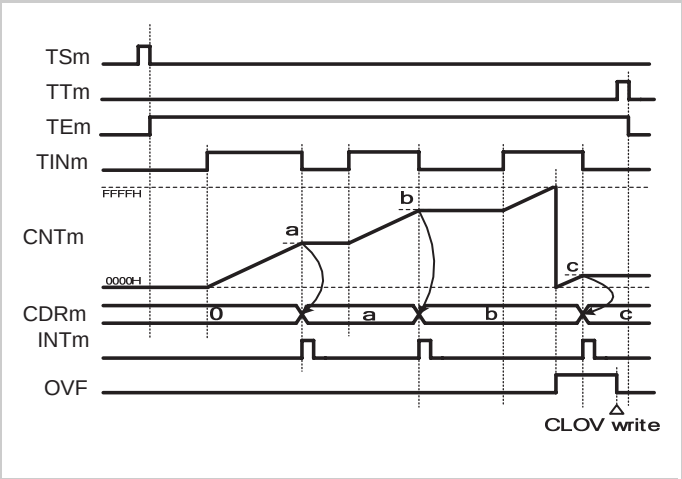


Figure 16-52 General timing diagram for TAUBnTTINm input period count detection function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-41 TAUBnCMORm settings for TAUBnTTINm input period count detection function

<R>

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	01: Overflow (TAUBnCSRm.OVF) set upon counter overflow and cleared by a CPU instruction (by setting the TAUBnCSCm.TAUBnCLOV bit to1).
MD[4:1]	1101: Capture & gate count mode
MD0	0: INTTAUBnIm not generated at operation start and start trigger during count disabled

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-42 TAUBnCMURm settings for TAUBnTTINm input period count detection function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm input period count detection function. Therefore, these registers must be set to 0.

Table 16-43 Simultaneous rewrite settings for TAUBnTTINm input period count detection function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for TAUBnTTINm input period count detection function

Table 16-44 Operating procedure for TAUBnTTINm input period count detection function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-41 "TAUBnCMORm settings for TAUBnTTINm input period count detection function" on page 1023* and *Table 16-42 "TAUBnCMURm settings for TAUBnTTINm input period count detection function" on page 1023*. Set the value of the TAUBnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. When a start edge is detected, TAUBnCNTm is cleared to 0000_H and TAUBnCNTm starts to count up.
During operation	Detection of TAUBnTTINm edges. The TAUBnCDRm, TAUBnCNTm, and TAUBnCSRm registers can be read at any time. The TAUBnCSC.CLOV bit can be set to 1.	When a TAUBnTTINm start edge (rising edge for high width measurement, falling edge for low width measurement) is detected, TAUBnCNTm starts to count up from the stop value. When TAUBnCNTm detects a capture edge (falling edge for high width measurement, rising edge for low width measurement), it transfers the value to TAUBnCDRm and INTTAUBnIm is generated. Counting stops at the "value transferred to TAUBnCDRm + 1" value and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. If the TAUBnCNTm reaches $FFFF_H$, the counter overflows and TAUBnCSR.OVF is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and it and TAUBnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

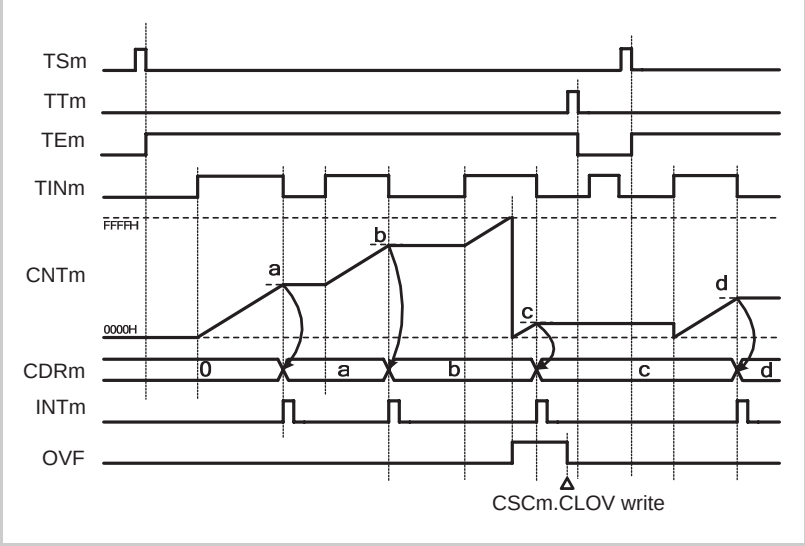


Figure 16-53 Operation stop and restart, TAUBnCMURm.TIS[1:0] = 11_B

- The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUBnTTINm input edges are ignored.
- The counter can be restarted by setting TAUBnTS.TSm to 1. TAUBnCNTm restarts to count from 0000_H.

16.15.5 Overflow interrupt output function (during TAUBnTTINm input period count detection)

(1) Overview

Summary This function measures the cumulative width of a TAUBnTTINm input signal. An interrupt is generated if the cumulative TAUBnTTINm input width is longer than $FFFF_H$.

Prerequisites

- The operation mode must be set to gate count mode, see *Table 16-45 "TAUBnCMORm settings for overflow interrupt output function (during TAUBnTTINm input period count detection)" on page 1029*.
- TAUBnTTOUTm is not used for this function.
- The value of TAUBnCDRm must be set to $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUBnTTINm input start edge is detected. $FFFF_H$ is written to TAUBnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value. The counter awaits the next TAUBnTTINm input start edge and then continues to count down from the current value.

When the counter reaches 0000_H an interrupt is generated. $FFFF_H$ is written to TAUBnCNTm and the counter continues to count down until a TAUBnTTINm input stop edge is detected.

Conditions The valid start and stop edges are specified by the TAUBnCMURm.TIS[1:0] bits:

- If TAUBnCMURm.TIS[1:0] = 10_B, the TAUBnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUBnCMURm.TIS[1:0] = 11_B, the TAUBnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

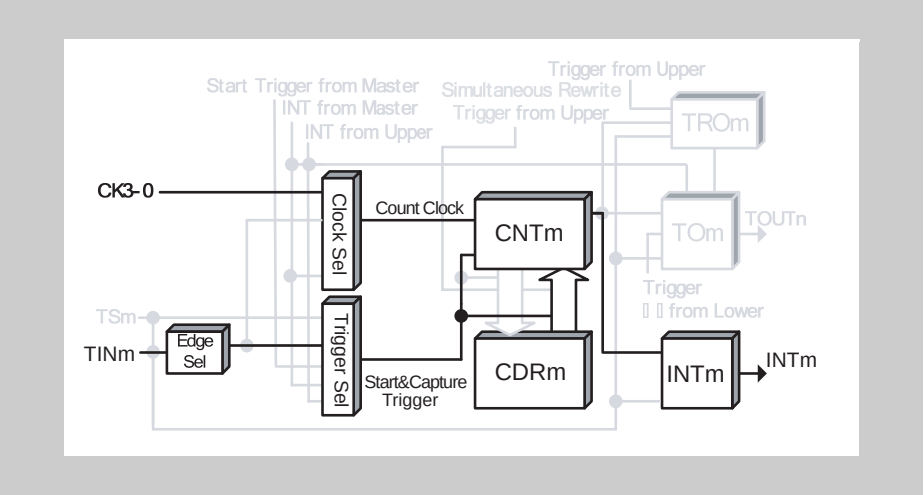


Figure 16-54 Block diagram for overflow interrupt output function (during TAUBnTTINm input period count detection)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUBnCMURm.TIS[1:0] = 11_B)

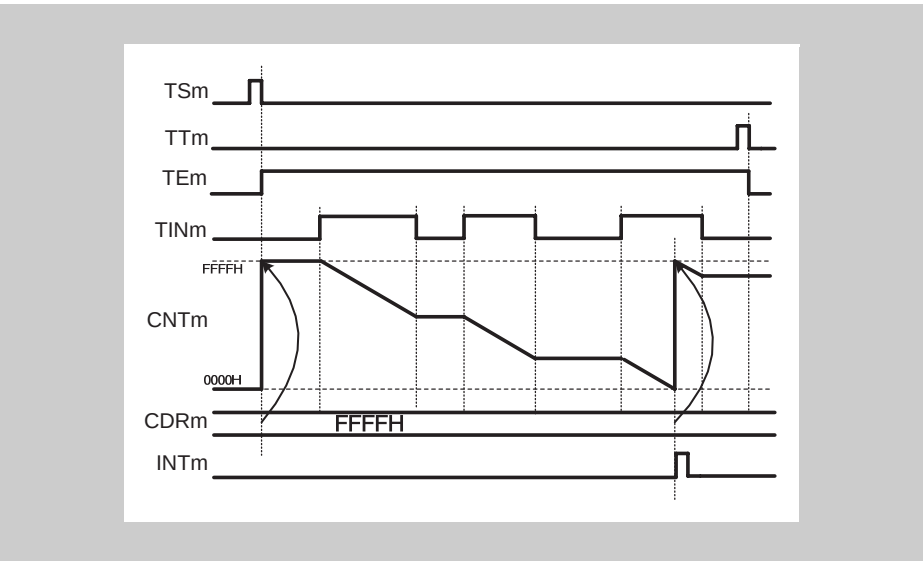


Figure 16-55 General timing diagram for overflow interrupt output function (during TAUBnTTINm input period count detection)

(3) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-45 TAUBnCMORm settings for overflow interrupt output function (during TAUBnTTINm input period count detection)

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1100: Gate count mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-46 TAUBnCMURm settings for overflow interrupt output function (during TAUBnTTINm input period count detection)

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the overflow interrupt output function (during TAUBnTTINm input period count detection). Therefore, these registers must be set to 0.

Table 16-47 Simultaneous rewrite settings for overflow interrupt output function (during TAUBnTTINm input period count detection)

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(4) Operating procedure for overflow interrupt output function (during TAUBnTTINm input period count detection)

Table 16-48 Operating procedure for overflow interrupt output function (during TAUBnTTINm input period count detection)

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-45 "TAUBnCMORm settings for overflow interrupt output function (during TAUBnTTINm input period count detection)" on page 1029* and *Table 16-46 "TAUBnCMURm settings for overflow interrupt output function (during TAUBnTTINm input period count detection)" on page 1029*. Set the value of the TAUBnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the start edge. When a start edge is detected, TAUBnCNTm loads the TAUBnCDRm value ($FFFF_H$).
During operation	The TAUBnCNTm register can be read at all times.	TAUBnCNTm counts down. When the counter reaches 0000_H: • INTTAUBnIm is generated • TAUBnCNTm loads the TAUBnCDRm value ($FFFF_H$) and continues to count down. When a reverse edge of TAUBnTTINm is detected during count operation: • TAUBnCNTm counts down from the stop value. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

16.15.6 TAUBnTTINm input pulse interval judgment function

(1) Overview

Summary This function outputs the result of a comparison between the count value (TAUBnCNTm) and the value in the channel data register (TAUBnCDRm) when a TAUBnTTINm input pulse occurs. An interrupt signal INTTAUBnIm is generated if the result of the comparison is true.

Prerequisites
- The operation mode must be set to judge mode, see *Table 16-49 "TAUBnCMORm settings for TAUBnTTINm input pulse interval judgment function" on page 1034*.
- TAUBnTTOUTm is not used for this function.

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value.

When a TAUBnTTINm valid edge is detected or TAUBnTS.TSm is set to 1, the function compares the current values of TAUBnCNTm and TAUBnCDRm. An interrupt signal INTTAUBnIm is generated if the result of the comparison is true. TAUBnCNTm reloads the value of TAUBnCDRm and subsequently continues operation, regardless of the result of the comparison.

If the counter reaches 0000_H before a TAUBnTTINm valid edge is detected, TAUBnCNTm overflows and is set to $FFFF_H$. It then continues to count down.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the function starts to count down.

Conditions The TAUBnCMORm.MD0 bit specifies the type of comparison:
- If TAUBnCMORm.MD0 = 0, INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm.
- If TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated when TAUBnCNTm > TAUBnCDRm.

(2) Block diagram and general timing diagram

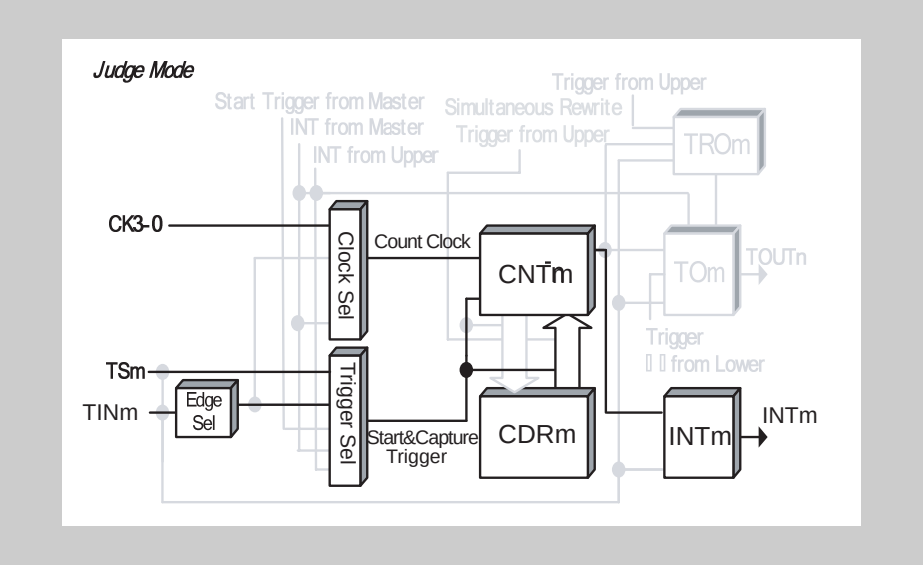


Figure 16-56 Block diagram for TAUBnTTINm input pulse interval judgment function

The following settings apply to the general timing diagram:

- INTTAUBnIm not generated at operation start (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

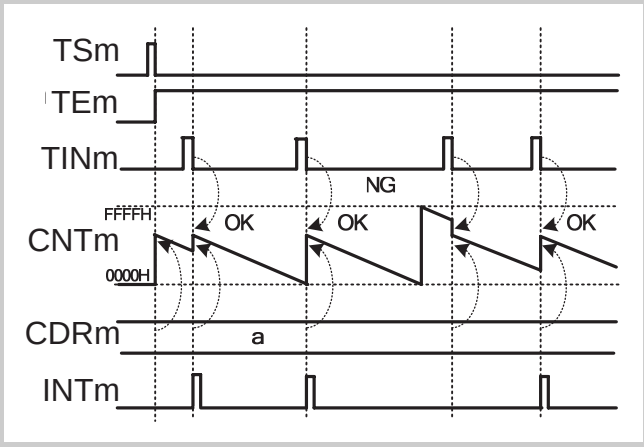


Figure 16-57 General timing diagram for TAUBnTTINm input pulse interval judgment function

(3) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-49 TAUBnCMORm settings for TAUBnTTINm input pulse interval judgment function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid edge of the TAUBnTTINm input signal is the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0001: Judge mode
MD0	0: INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm 1: INTTAUBnIm is generated when TAUBnCNTm > TAUBnCDRm

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-50 TAUBnCMURm settings for TAUBnTTINm input pulse interval judgment function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

<R>

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm input pulse interval judgment function. Therefore, these registers must be set to 0.

Table 16-51 Simultaneous rewrite settings for TAUBnTTINm input pulse interval judgment function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(4) Operating procedure for for TAUBnTTINm input pulse interval judgment function

Table 16-52 Operating procedure for TAUBnTTINm input pulse interval judgment function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-49 "TAUBnCMORm settings for TAUBnTTINm input pulse interval judgment function" on page 1034* and *Table 16-50 "TAUBnCMURm settings for TAUBnTTINm input pulse interval judgment function" on page 1034*. Set the value of the TAUBnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value.
During operation	Detection of TAUBnTTINm edges. The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at any time.	TAUBnCNTm counts down. When a TAUBnTTINm input edge is detected • TAUBnCNTm reloads TAUBnCDRm and continues count operation. • TAUBnCNTm compares the values and judges the condition according to the TAUBnCMORm.MD0 setting. • If the condition is satisfied, INTTAUBnIm is generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

16.15.7 TAUBnTTINm input signal width judgment function

(1) Overview

Summary This function outputs the result of a comparison between the count value (TAUBnCNTm) and the value in the channel data register (TAUBnCDRm) at a valid stop edge of a TAUBnTTINm input signal. An interrupt signal INTTAUBnIm is generated if the result of the comparison is true.

Prerequisites

- The operation mode must be set to Judge & one-count mode, see *Table 16-53 "TAUBnCMORm settings for TAUBnTTINm input signal width judgment function" on page 1039*
- TAUBnTTOUTm is not used for this function

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. When a valid TAUBnTTINm input start edge is detected, the current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value.

When a TAUBnTTINm valid stop edge is detected, the function compares the current values of TAUBnCNTm and TAUBnCDRm. An interrupt signal INTTAUBnIm is generated if the result of the comparison is true. The counter TAUBnCNTm retains its value until the next TAUBnTTINm valid start edge is detected, regardless of the result of the comparison.

If the counter reaches 0000_H before a valid TAUBnTTINm stop edge is detected, TAUBnCNTm overflows and is set to $FFFF_H$. It then continues to count down.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the function starts to count down.

Conditions

- The TAUBnCMORm.MD0 bit specifies the type of comparison:
  - If TAUBnCMORm.MD0 = 0, INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm.
  - If TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated when TAUBnCNTm > TAUBnCDRm.
- The TAUBnCMURm.TIS[1:0] bits specify the type of width measurement:
  - For high width measurement, the start edge is a rising TAUBnTTINm edge and the stop edge is a falling TAUBnTTINm edge.
  - For low width measurement, the start edge is a falling TAUBnTTINm edge and the stop edge is a rising TAUBnTTINm edge.
- Forced restart is not possible for this function.

(2) Block diagram and general timing diagram

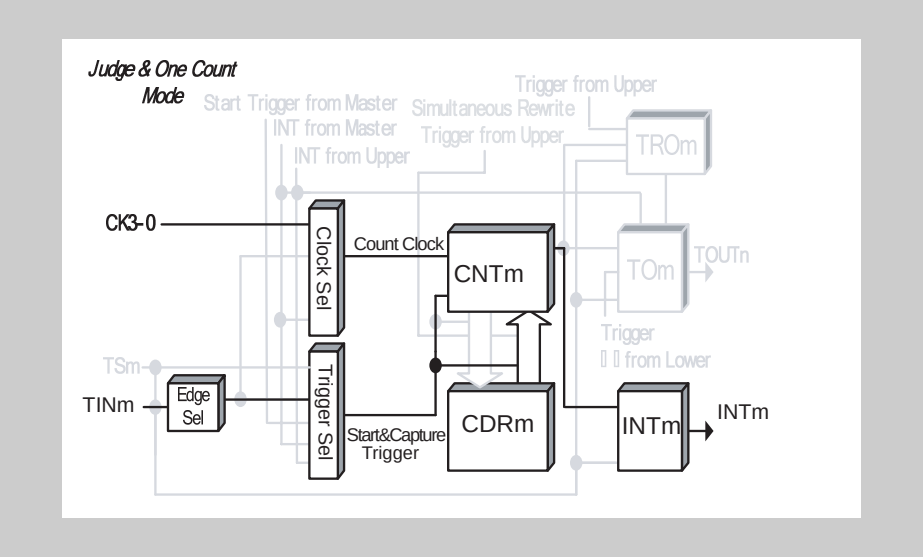


Figure 16-58 Block diagram for TAUBnTTINm input signal width judgment function

The following settings apply to the general timing diagram:

- INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm (TAUBnCMORm.MD0 = 0)
- TAUBnTTINm valid start edge = rising edge, TAUBnTTINm valid stop edge = falling edge (TAUBnCMURm.TIS[1:0] = 11_B)

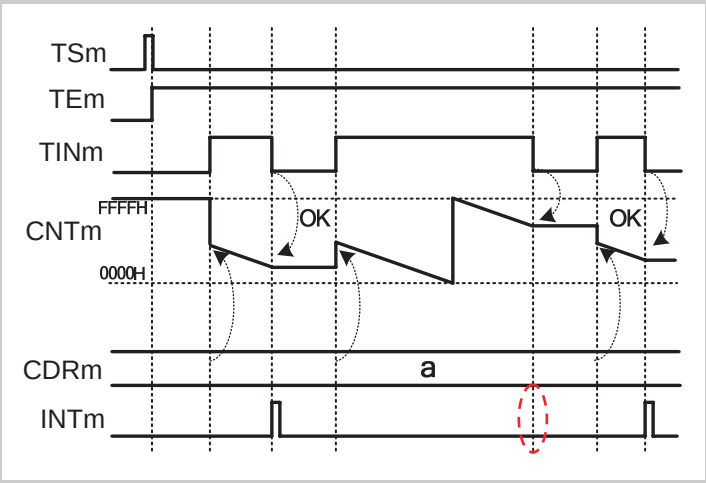


Figure 16-59 General timing diagram for TAUBnTTINm input signal width judgment function

(3) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-53 TAUBnCMORm settings for TAUBnTTINm input signal width judgment function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0111: Judge & one-count mode
MD0	0: INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm 1: INTTAUBnIm is generated when TAUBnCNTm > TAUBnCDRm

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-54 TAUBnCMURm settings for TAUBnTTINm input signal width judgment function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm input signal width judgment function. Therefore, these registers must be set to 0.

Table 16-55 Simultaneous rewrite settings for TAUBnTTINm input signal width judgment function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(4) Operating procedure for TAUBnTTINm input signal width judgment function

Table 16-56 Operating procedure for TAUBnTTINm input signal width judgment function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-53 "TAUBnCMORm settings for TAUBnTTINm input signal width judgment function" on page 1039* and *Table 16-54 "TAUBnCMURm settings for TAUBnTTINm input signal width judgment function" on page 1039*. Set the value of the TAUBnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. When a TAUBnTTINm start edge is detected, TAUBnCNTm loads the TAUBnCDRm value.
During operation	Detection of TAUBnTTINm edges The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at any time.	TAUBnCNTm counts down. When a TAUBnTTINm stop edge is detected: • TAUBnCNTm stops and waits for detection of the TAUBnTTINm start edge. • TAUBnCNTm compares the values and judges the condition according to the TAUBnCMORm.MD0. • If the condition is satisfied, INTTAUBnIm is generated Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

16.16 Independent Channel Simultaneous Rewrite Functions

This chapter describes functions that carry out simultaneous rewrite:

- *16.16.1 “Simultaneous rewrite trigger generation function type 1”*

16.16.1 Simultaneous rewrite trigger generation function type 1

(1) Overview

Summary This function generates an interrupt on a specific channel that can be used by lower channels as a simultaneous rewrite trigger. The interrupt is generated at regular intervals.

Prerequisites

- Two (or more) channels, each with simultaneous rewrite enabled (TAUBnRDE.RDEm = 1).
- The operation mode of the upper channel must be set to interval timer mode, see *Table 16-57 "TAUBnCMORm settings for simultaneous rewrite trigger generation function type 1" on page 1046*.
- The operation mode of the lower channel(s) can be set as desired.

Description The counters are started by setting the channel trigger bits (TAUBnTS.TSm and TAUBnTS.TSm_lower) to 1. This in turn sets TAUBnTE.TEm and TAUBnTE.TEm_lower = 1, enabling count operation. The current value of the data register buffer of the upper channel (TAUBnCDRm buf) is written to the counter (TAUBnCNTm) and the counter starts to count down from this value. The counter(s) of the lower channel(s) start to count as specified by their selected operating modes.

When a counter reaches 0000_H, an interrupt is generated from the channel. The corresponding TAUBnCNTm then reloads the current TAUBnCDRm buffer value and subsequently continues operation.

If the channel where the interrupt occurs is specified as the trigger channel for simultaneous rewrite (TAUBnRDC.RDCm = 1) and is an upper channel, simultaneous rewrite takes place on all lower channels in which simultaneous rewrite is currently possible (TAUBnRSF.RSFm = 1).

The values of the data registers are copied to the corresponding data register buffers. Each time a counter starts to count down, it reads the value in the data register buffer and counts down from this value.

The value of a data register can be changed at any time, but it is only transferred to the corresponding data register buffer when simultaneous rewrite occurs.

Conditions

- The channel which is monitored for INTTAUBnIm is specified by setting TAUBnRDC.RDCm = 1 for the corresponding channel. The TAUBnRDC.RDCm bit must be 0 for all other channels in which simultaneous rewrite should take place.
- If the TAUBnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details, see *16.10 "TAUBnTTOUTm Toggle and INTTAUBnIm Generation When Counter Start Is Triggered (MD0 Bit)" on page 971*.

(2) Equations

Simultaneous rewrite trigger generation cycle = count clock cycle × (TAUBnCDRm + 1)

To control simultaneous rewrite, the following condition must be satisfied:

TAUBnCDRm = [(value of TAUBnCDRm of master channel subject to simultaneous rewrite + 1) × number of interrupts] - 1

That is, the ratio of TAUBnCDRm + 1 and TAUBnCDRm_master + 1 must be an integer. This integer corresponds to the number of interrupts.

(3) Block diagram and general timing diagram

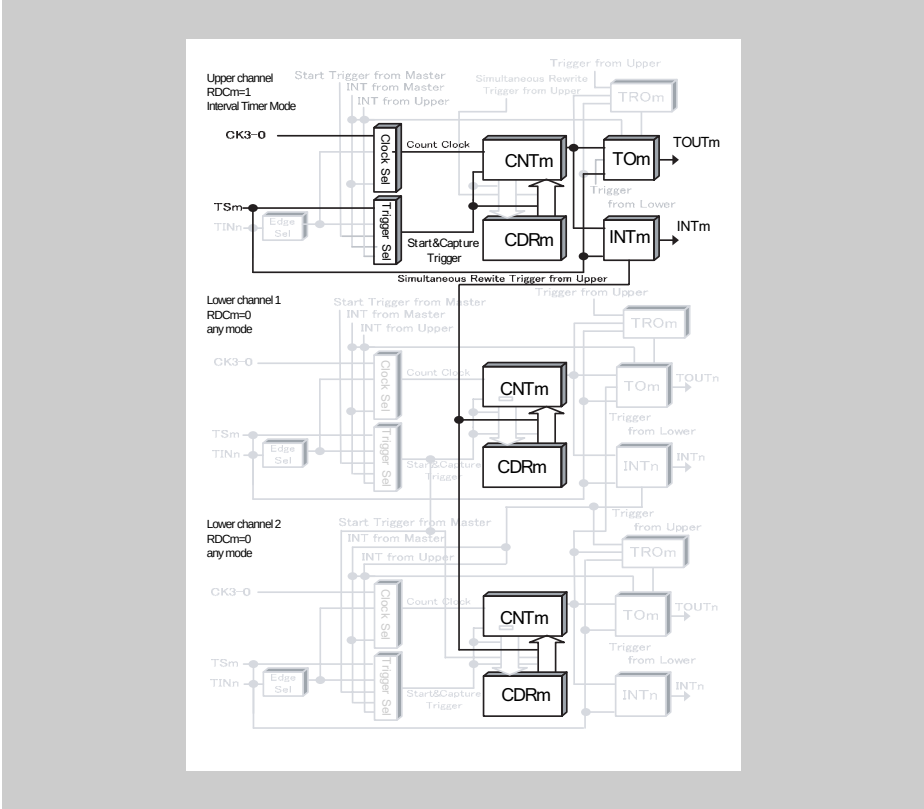


Figure 16-60 Block diagram for simultaneous rewrite trigger generation function type 1

The following settings apply to the general timing diagram:

- INTTAUBnIm generated at operation start (TAUBnCMORm.MD0 = 1)

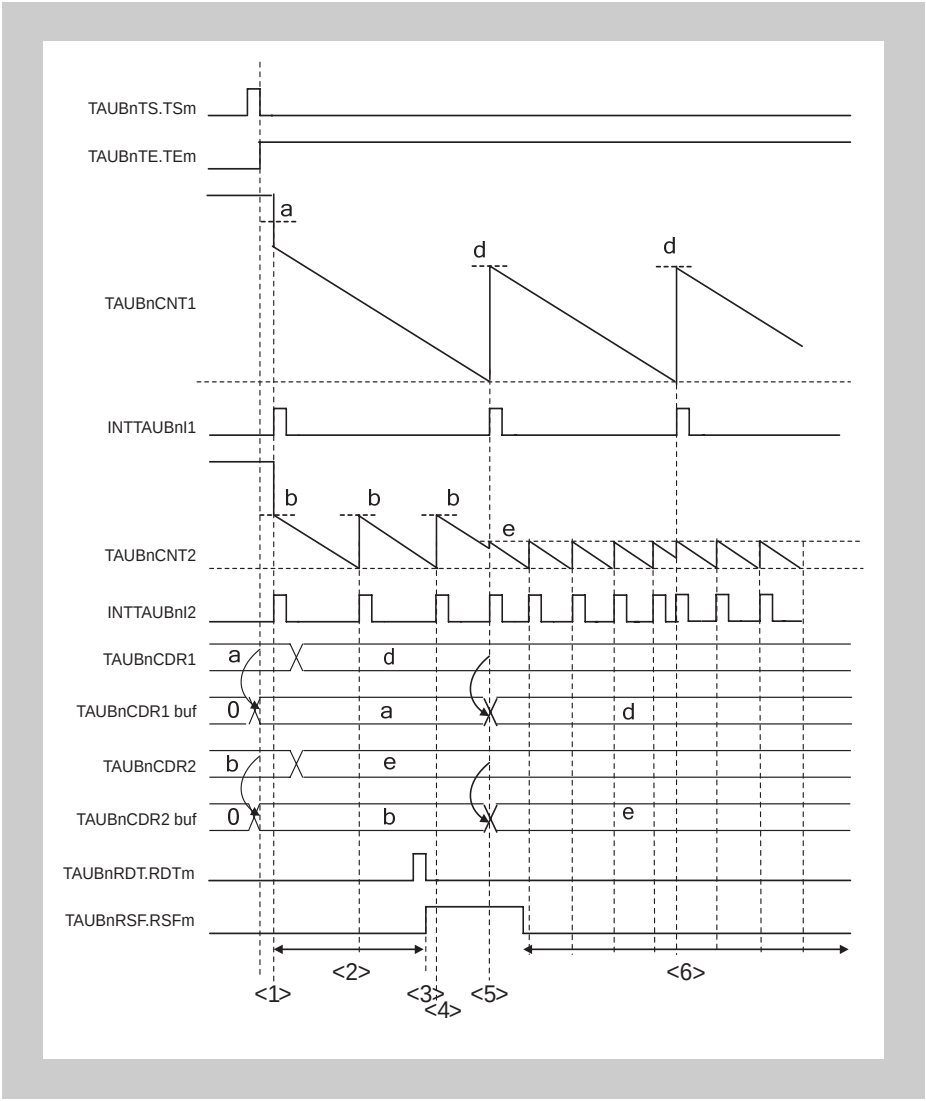


Figure 16-61 General timing diagram for simultaneous rewrite trigger generation function type 1

(4) Register settings for the upper channel

(a) TAUBnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-57 TAUBnCMORm settings for simultaneous rewrite trigger generation function type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval timer mode
MD0	0: INTTAUBnIm not generated at operation start 1: Generates INTTAUBnIm at operation start

(b) TAUBnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-58 TAUBnCMURm settings for simultaneous rewrite trigger generation function type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode for the upper channel

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the upper channel

Table 16-59 Simultaneous rewrite settings for simultaneous rewrite trigger generation function type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDM.RDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
RDC.RDCm	1: Channel is monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger

(5) Register settings for the lower channel(s)

(a) TAUBnCMORm for the lower channel(s)

The TAUBnCMORm register of the lower channel(s) can be set arbitrarily.

(b) TAUBnCMURm for the lower channel(s)

The TAUBnCMURm register of the lower channel(s) can be set arbitrarily.

(c) Channel output mode for the lower channel(s)

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the lower channel(s)

Table 16-60 Simultaneous rewrite settings for the lower channel in simultaneous rewrite trigger generation function type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDM.RDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous-rewrite trigger

(6) Operating procedure for simultaneous rewrite trigger generation function type 1

Table 16-61 Operating procedure for simultaneous rewrite trigger generation function type 1

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers for the upper channel as described in *Table 16-57 "TAUBnCMORm settings for simultaneous rewrite trigger generation function type 1" on page 1046* and *Table 16-58 "TAUBnCMURm settings for simultaneous rewrite trigger generation function type 1" on page 1046*. Set the TAUBnCMORm register and TAUBnCMURm registers for the lower channel as described in *5 "Register settings for the lower channel(s)"*. Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value. When TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated and TAUBnTTOUTm toggles.
During operation	TAUBnRDT.RDTm can be changed. TAUBnRSF.RSFm can be read at all times.	TAUBnCNTm counts down. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated • TAUBnTTOUTm toggles. Simultaneous rewrite is controlled when INTTAUBnIm is generated from the channel where TAUBnRDC.RDCm is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnTTOUTm retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

16.17 Other Independent Channel Functions

This chapter describes a function that generates an interrupt when a certain number of TAUBnTTINm pulses has occurred, a function that divides the frequency of TAUBnTTINm, and a function that measures the duration between the function start and a TAUBnTTINm input signal:

- *16.17.1 “External event count function”*
- *16.17.2 “Clock divide function”*
- *16.17.3 “TAUBnTTINm input position detection function”*

16.17.1 External event count function

(1) Overview

Summary This function is used as an event timer. It generates an interrupt (INTTAUBnIm) when a specific number of TAUBnTTINm input pulses has occurred.

Prerequisites
- The operation mode must be set to event count mode, see *Table 16-62 "TAUBnCMORm settings for external event count function" on page 1052*.
- TAUBnTTOUTm is not used for this function.

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. When the counter starts, the current value of TAUBnCDRm is written to TAUBnCNTm.

When a valid TAUBnTTINm input edge is detected, the value of TAUBnCNTm reduces by 1. TAUBnCNTm retains this value until a valid TAUBnTTINm input edge is detected or the counter is restarted.

When the counter value reaches 0000_H, INTTAUBnIm is generated. TAUBnCNTm then reloads the TAUBnCDRm value and subsequently continues operation.

The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm stops and retains its value. The counter can be restarted by setting TAUBnTS.TSm to 1. The counter can also be restarted without stopping it first (forced restart) by setting TAUBnTS.TSm to 1 during operation.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUBnCMURm.TIS[1:0] bits:
- If TAUBnCMURm.TIS[1:0] = 00_B, the falling edges are counted.
- If TAUBnCMURm.TIS[1:0] = 01_B, the rising edges are counted.
- If TAUBnCMURm.TIS[1:0] = 10_B, the rising and falling edges are counted.

(2) Equations

Number of valid edges,
detected before INTTAUBnIm is generated = TAUBnCDRm + 1

(3) Block diagram and general timing diagram

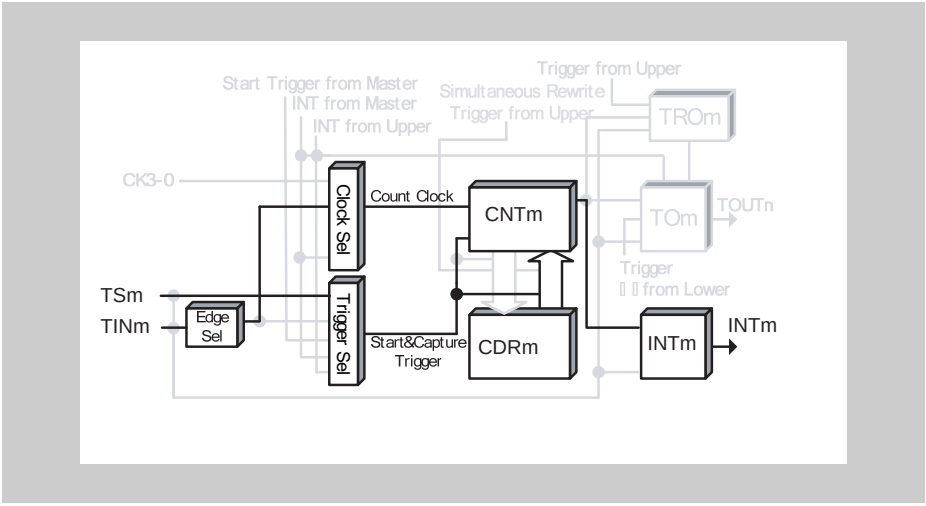


Figure 16-62 Block diagram for external event count function

The following settings apply to the general timing diagram:

- Rising edge detection (TAUBnCMURm.TIS[1:0] = 01_B)

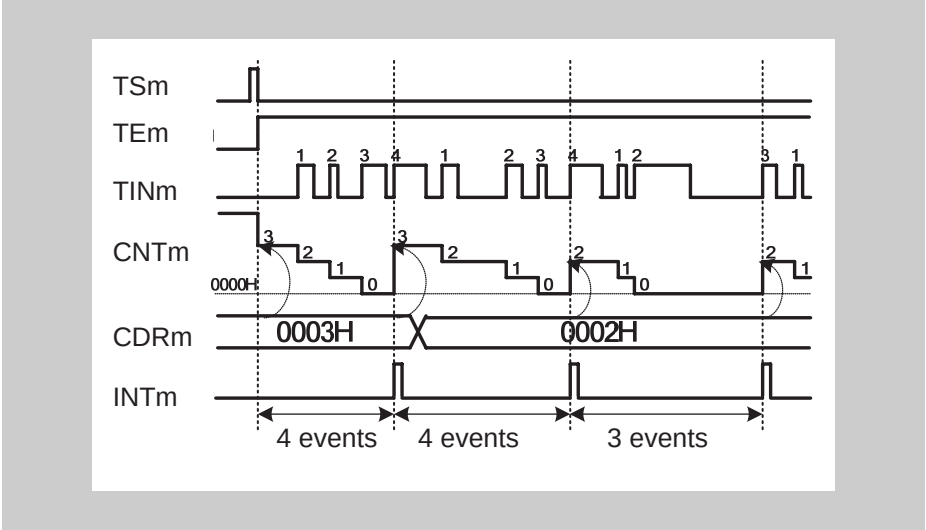


Figure 16-63 General timing diagram for external event count function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-62 TAUBnCMORm settings for external event count function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	1: Valid TAUBnTTINm input edge is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0011: Event count mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-63 TAUBnCMURm settings for external event count function

Bit name	Setting
TIS[1:0]	00: Falling edge 01: Rising edge 10: Rising and falling edge

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the external event count function. Therefore, these registers must be set to 0.

Table 16-64 Simultaneous rewrite settings for external event count function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for external event count function

Table 16-65 Operating procedure for external event count function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-62 "TAUBnCMORm settings for external event count function" on page 1052* and *Table 16-63 "TAUBnCMURm settings for external event count function" on page 1052*. Set the value of the TAUBnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value and waits for detection of the TAUBnTTINm input edge.
During operation	Detection of TAUBnTTINm edges. The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at any time.	TAUBnCNTm performs count-down operation each time a TAUBnTTINm input edge is detected. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

(6) Specific timing diagrams

(a) TAUBnCDRm = 0000_H

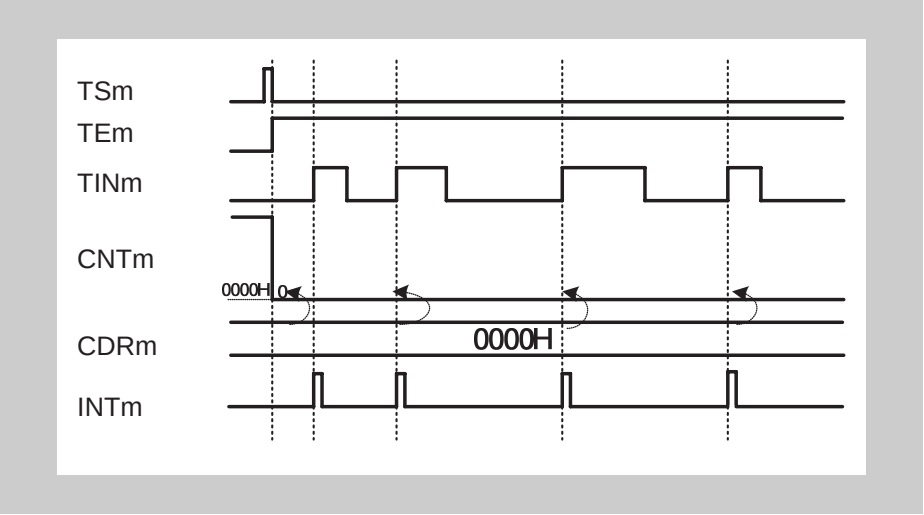


Figure 16-64 TAUBnCDRm = 0000H, TAUBnCMURm.TIS[1:0] = 01

- If 0000_H = TAUBnCDRm, 0000_H is written to TAUBnCNTm every time a valid TAUBnTTINm input edge is detected.

  This means, INTTAUBnIm is generated every time a valid TAUBnTTINm input edge is detected.

(b) Operation stop and restart

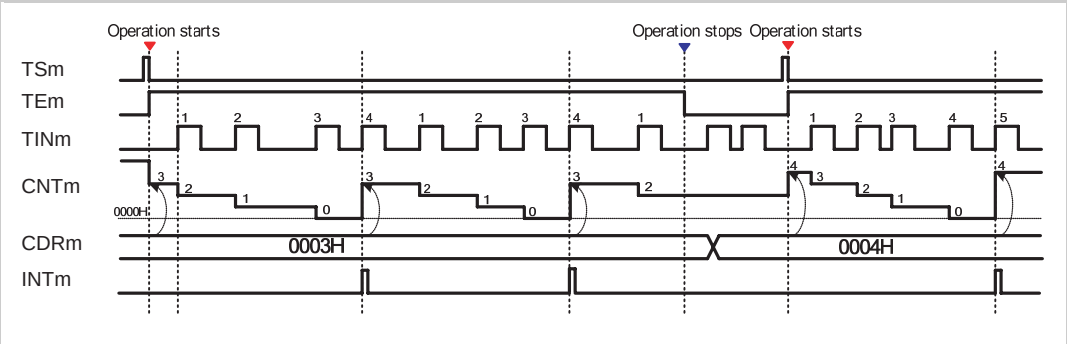


Figure 16-65 Operation stop and restart, TAUBnCMURm.TIS[1:0] = 01

- The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm stops and the current value is retained. TAUBnTTINm continues and TAUBnCNTm ignores the valid edge.
- The counter can be restarted by setting TAUBnTS.TSm to 1. TAUBnCNTm loads the TAUBnCDRm value and restarts count operation.

(c) Forced restart

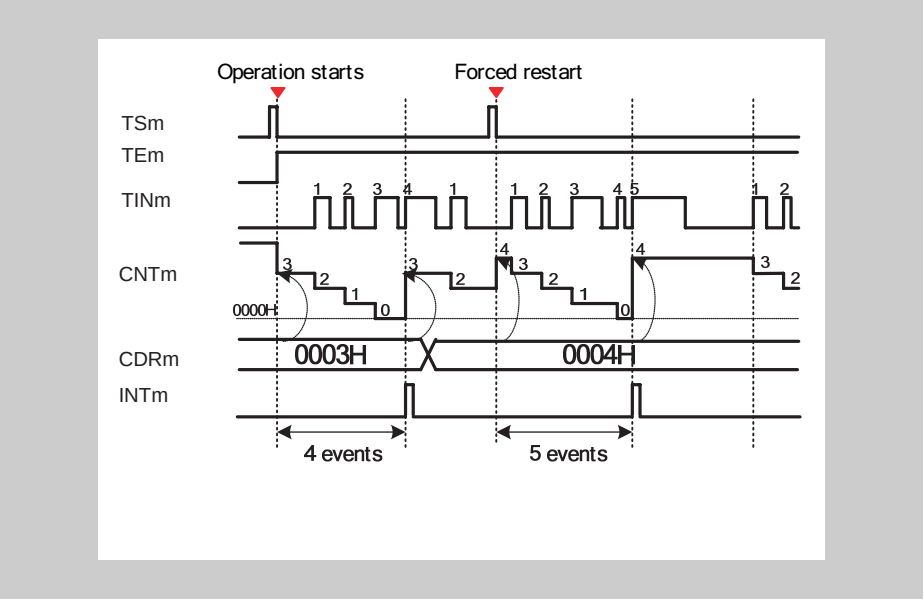


Figure 16-66 Forced restart, TAUBnCMURm.TIS[1:0] = 01

A forced restart applies a change to TAUBnCDRm immediately.

- The counter can be restarted (without stopping it first), by setting TAUBnTS.TSm to 1 during operation.
- The value of TAUBnCDRm is written to TAUBnCNTm and the counter awaits the next valid TAUBnTTINm input edge.

16.17.2 Clock divide function

(1) Overview

Summary This function is used as a frequency divider. The frequency of the input signal TAUBnTTINm is divided by a factor related to TAUBnCDRm, and the resulting signal is output to TAUBnTTOUTm.

Prerequisites

- TAUBnTTINm must have a fixed frequency
- The operation mode must be set to interval timer mode, see *Table 16-66 "TAUBnCMORm settings for clock divide function" on page 1059*.
- The channel output mode must be set to independent channel output mode 1, see *16.8 "Channel Output Modes" on page 962*.

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value, using TAUBnTTINm as the count clock.

When the counter value reaches 0000_H, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles. TAUBnCNTm then reloads the TAUBnCDRm value and subsequently continues operation.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the function starts to count down.

The counter can be stopped by setting TAUBnTT.TTm = 1, which in turn sets TAUBnTE.TEm = 0. TAUBnCNTm and TAUBnTTOUTm stop but retain their values. The function can be restarted by setting TAUBnTS.TSm = 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm = 1 during operation.

Conditions If the TAUBnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUBnTTOUTm does not toggle. This results in an inverted TAUBnTTOUTm signal compared to when TAUBnCMORm.MD0 is set to 1. For details, see *16.10 "TAUBnTTOUTm Toggle and INTTAUBnIm Generation When Counter Start Is Triggered (MD0 Bit)" on page 971*.

Note The input TAUBnTTINm is sampled at the frequency of the operation clock, specified by TAUBnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUBnTTOUTm has an error of ± 1 operation clock cycle.

(2) Equations

- When rising edge detection is selected:
  TAUBnTTOUTm frequency = TAUBnTTINm frequency / [(TAUBnCDRm + 1) x 2]
- When falling edge detection is selected:
  TAUBnTTOUTm frequency = TAUBnTTINm frequency / [(TAUBnCDRm + 1) x 2]
- When rising and falling edge detection is selected:
  TAUBnTTOUTm frequency = TAUBnTTINm frequency / (TAUBnCDRm + 1)

(3) Block diagram and general timing diagram

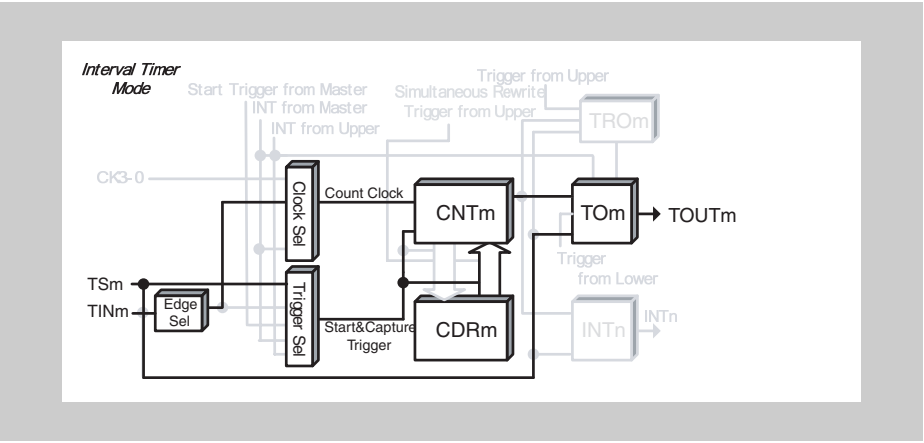


Figure 16-67 Block diagram for clock divide function

The following settings apply to the general timing diagram:

- INTTAUBnIm generated at operation start (TAUBnCMORm.MD0 = 1)
- Rising edge detection (TAUBnCMURm.TIS[1:0] = 01_B)

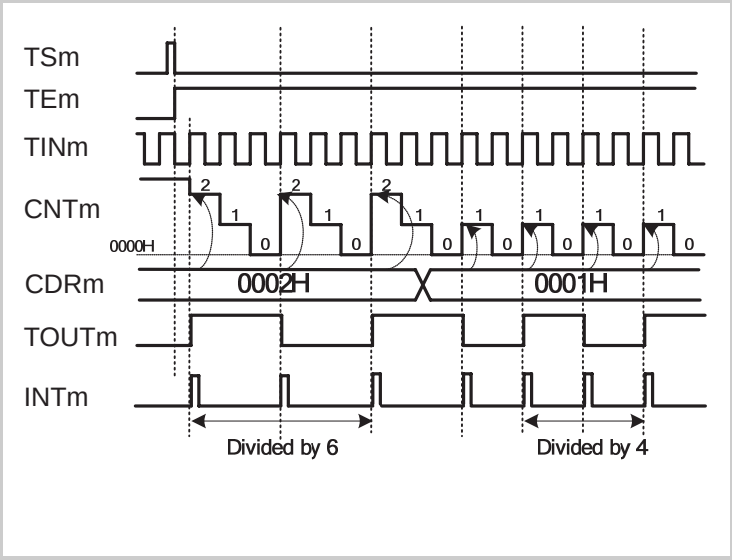


Figure 16-68 General timing diagram for clock divide function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-66 TAUBnCMORm settings for clock divide function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	1: Valid TAUBnTTINm input edge is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval timer mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-67 TAUBnCMURm settings for clock divide function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Table 16-68 Control bit settings for independent channel output mode 1

Bit name	Setting
TOE.TOEm	1: Disables independent channel output mode controlled by software
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDL.TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

Note The channel output mode can also be set to channel output mode controlled by software by setting TAUBnTOE.TOEm = 0. TAUBnTTOUTm can then be controlled independently of the interrupts. For details, see *Table 16-10 “Channel output modes” on page 963*.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the clock divide function. Therefore, these registers must be set to 0.

Table 16-69 Simultaneous rewrite settings for clock divide function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for clock divide function

Table 16-70 Operating procedure for clock divide function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-66 "TAUBnCMORm settings for clock divide function" on page 1059* and *Table 16-67 "TAUBnCMURm settings for clock divide function" on page 1059*. Set the value of the TAUBnCDRm register. Set the channel output mode by setting the control bits as described in *Table 16-68 "Control bit settings for independent channel output mode 1" on page 1060*.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value. When TAUBnCMORm.MD0 is set to 1, INTTAUBnIm is generated and TAUBnTTOUTm toggles.
During operation	The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at all times.	When a TAUBnTTINm input edge is detected, TAUBnCNTm counts down. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated • TAUBnTTOUTm toggles. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnTTOUTm retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

(6) Specific timing diagrams

(a) TAUBnCDRm = 0000_H

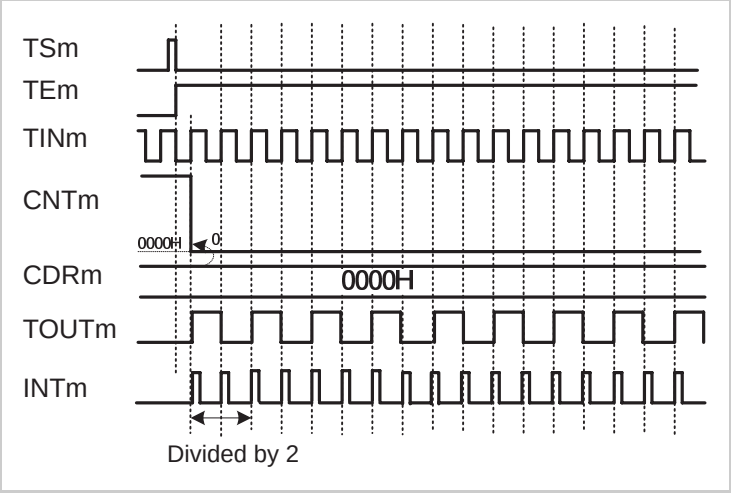


Figure 16-69 TAUBnCDRm = 0000_H, TAUBnCMORm.MD0 = 1, TAUBnCMURm.TIS[1:0] = 01

- If TAUBnCDRm is 0000_H, TAUBnCNTm is also always 0000_H.
- INTTAUBnIm is generated every count clock, resulting in TAUBnTTOUTm toggling every count clock.

(b) Restart

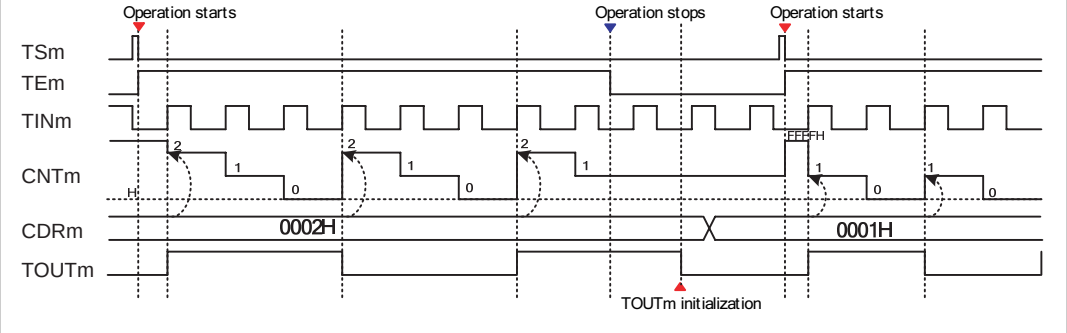


Figure 16-70 Restart, TAUBnCMORm.MD0 = 1, TAUBnCMURm.TIS[1:0] = 01

To reset the value of TAUBnTTOUTm:

- Set TAUBnTOE.TOEm = 0 when the counter is stopped (TAUBnTE.TEm = 0)
- Then write either 0 or 1 to TAUBnTO.TOm to set the new start value of TAUBnTTOUTm

(c) Forced restart

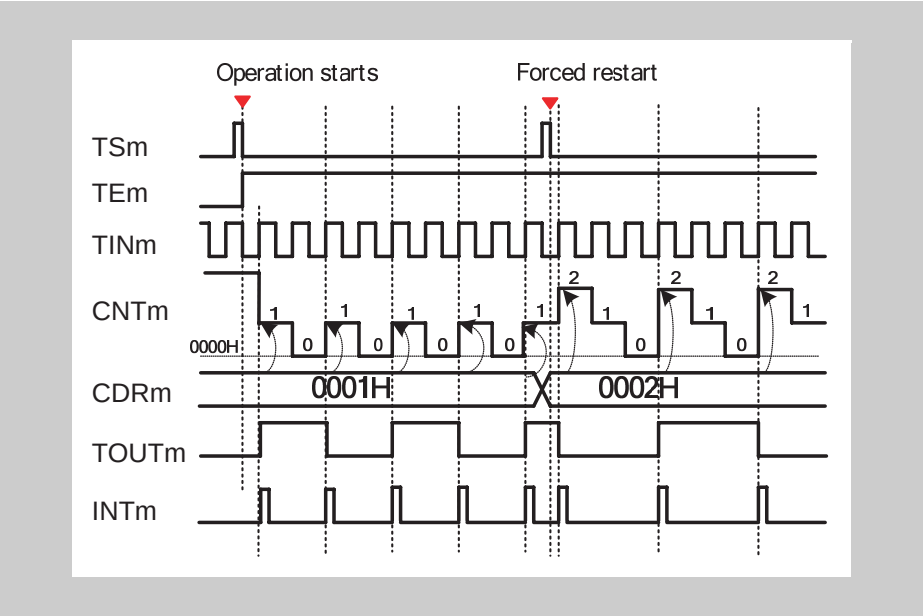


Figure 16-71 Forced restart, TAUBnCMORm.MD0 = 1, TAUBnCMURm.TIS[1:0] = 01

- The counter can be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm = 1 during operation.
- The value of TAUBnCDRm is written to TAUBnCNTm and the count operation restarts.
- TAUBnTTOUTm restarts at the same level as before the forced restart.

16.17.3 TAUBnTTINm input position detection function

(1) Overview

Summary This function measures the duration between the function start and a TAUBnTTINm input signal.

Prerequisites
- The operation mode must be set to Count capture mode, see *Table 16-71 "TAUBnCMORm settings for TAUBnTTINm input position detection function" on page 1066*.
- TAUBnTTOUTm is not used for this function.

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The counter starts to count from 0000_H. When a valid TAUBnTTINm input stop edge is detected, the current TAUBnCNTm value is written to TAUBnCDRm and an interrupt (INTTAUBnIm) is generated. The counter continues to count from the current value until the next valid TAUBnTTINm input edge is detected.

When the counter reaches $FFFF_H$, the bit TAUBnCSRm.OVF is set to 1 and the counter restarts from 0000_H. The value of TAUBnCSRm.OVF is reset by the CPU by TAUBnCSCm.CLOV = 1.

Note The input TAUBnTTINm is sampled at the frequency of the operation clock, specified by the TAUBnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUBnTTOUTm has an error of ± 1 operation clock cycle.

Conditions If the TAUBnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details, see *16.10 "TAUBnTTOUTm Toggle and INTTAUBnIm Generation When Counter Start Is Triggered (MD0 Bit)" on page 971*.

(2) Equations

Function duration at a TAUBnTTINm input pulse =

<R> count clock cycle × [(($FFFF_H$ + 1) × TAUBnCSRm.OVF) + (TAUBnCDRm capture value + 1)]

(3) Block diagram and general timing diagram

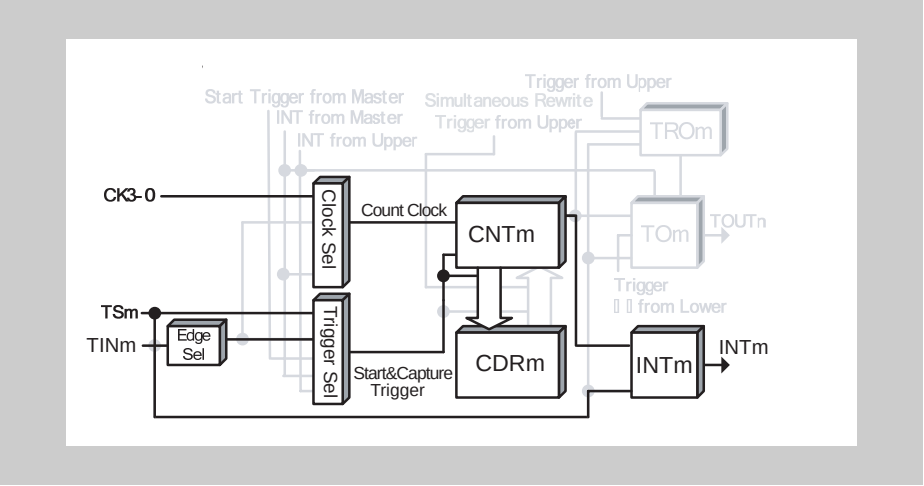


Figure 16-72 Block diagram for TAUBnTTINm input position detection function

The following settings apply to the general timing diagram:

- INTTAUBnIm not generated at operation start (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

<R>

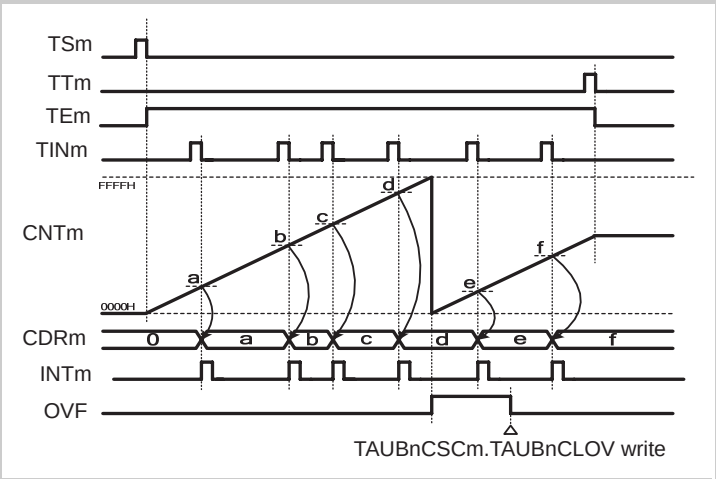


Figure 16-73 General timing diagram for TAUBnTTINm input position detection function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-71 TAUBnCMORm settings for TAUBnTTINm input position detection function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUBnTTINm input edge signal is used as the external capture trigger
COS[1:0]	01: Overflow (TAUBnCSRm.OVF) set upon counter overflow and cleared by a CPU instruction (by setting the TAUBnCSCm.TAUBnCLOV bit to 1).
MD[4:1]	1011: Count capture mode
MD0	0: INTTAUBnIm not generated at operation start 1: Generates INTTAUBnIm at operation start

<R>

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-72 TAUBnCMURm settings for TAUBnTTINm input position detection function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm input position detection function. Therefore, these registers must be set to 0.

Table 16-73 Simultaneous rewrite settings for TAUBnTTINm input position detection function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for TAUBnTTINm input position detection function

Table 16-74 Operating procedure for TAUBnTTINm input position detection function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-71 "TAUBnCMORm settings for TAUBnTTINm input position detection function" on page 1066* and *Table 16-72 "TAUBnCMURm settings for TAUBnTTINm input position detection function" on page 1066*. Set the value of the TAUBnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. INTTAUBnIm is generated when TAUBnCMORm.MD0 is set to 1.
During operation	The TAUBnCMURm.TIS[1:0] bits can be changed at any time. The TAUBnCDRm and TAUBnCSRm registers can be read at any time.	TAUBnCNTm starts to count up from 0000_H. When a TAUBnTTINm valid edge is detected: • TAUBnCNTm transfers (captures) its value to TAUBnCDRm • TAUBnTTINm is output. • The counter value is not cleared to 0000_H and TAUBnCNTm continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

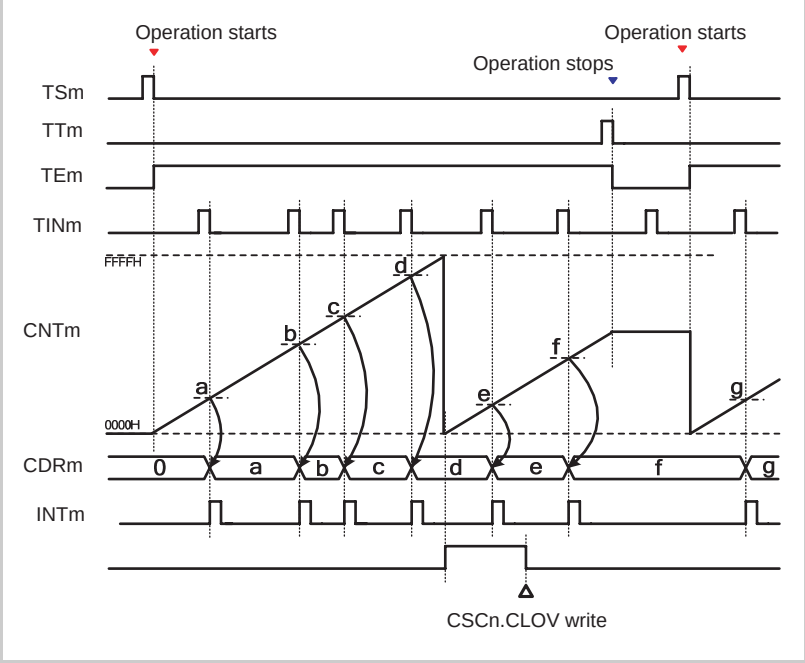


Figure 16-74 Operation stop and restart, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0] = 00

- The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUBnTTINm input edges are ignored.
- The counter can be restarted by setting TAUBnTS.TSm to 1. TAUBnCNTm restarts to count from 0000_H.

16.18 Synchronous Channel Operation Functions

This section lists all the synchronous channel operation functions provided by the Timer Array Unit B. For a general overview of synchronous channel operation, see *16.3 “Functional Description” on page 946*.

16.19 Synchronous PWM Signal Functions Triggered at Regular Intervals

This chapter describes functions that generate PWM signals at regular intervals, that can be reset by a TAUBnTTINm input, with and without dead time.

- *16.19.1 “PWM output function”*
- *16.19.2 “Delay pulse output function”*
- *16.19.3 “AD conversion trigger output function type 1”*

16.19.1 PWM output function

(1) Overview

Summary This function generates multiple PWM outputs by using a master and multiple slave channels. It enables the pulse cycle (frequency) and the pulse width (duration) of the TAUBnTTOUTm to be set. The pulse cycle is set in the master channel. The pulse width is set in the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to interval timer mode, see *Table 16-75 “TAUBnCMORm settings for the master channel of the PWM output function” on page 1074*.
- The operation mode of the slave channel(s) must be set to one-count mode, see *Table 16-78 “TAUBnCMORm settings for the slave channel of the PWM output function” on page 1076*.
- TAUBnTTOUTm is not used for the master channel of this function.
- The channel output mode of the slave channel(s) must be set to synchronous channel output mode 1 (*16.8 “Channel Output Modes” on page 962*).

Description The counters are started by setting the channel trigger bits (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The current value of TAUBnCDRm is written to TAUBnCNTm and the counters start to count down from these values. INTTAUBnIm is generated on the master channel and TAUBnTTOUTm (slave) toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed and INTTAUBnIm is generated. The counter reloads the TAUBnCDRm value and counts down.

- Slave channel(s)

  The INTTAUBnIm of the master channel triggers the counter of the slave channel(s). The current value of TAUBnCDRm (slave) is written to TAUBnCNTm (slave) and the counter starts to count down from this value. The TAUBnTTOUTm signal is set.

  When the counter reaches 0000_H, i.e. duty time has elapsed, INTTAUBnIm is generated and the TAUBnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUBnIm of the master channel, and thus the start of the next pulse cycle.

The counter can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channel(s) stop but retain their values. The counters can be restarted by setting TAUBnTS.TSm to 1.

Note If a forced restart is executed during operation, the counter value becomes invalid and TAUBnTTOUTm is not output as expected PWM signal.

Conditions Simultaneous rewrite can be used with this function. See *16.7 “Simultaneous Rewrite” on page 952*.

(2) Equations

Pulse cycle = (TAUBnCDRm (master) + 1) x count clock cycle

Duty cycle [%] = (TAUBnCDRm (slave) / (TAUBnCDRm (master) + 1)) x 100

– Duty cycle = 0 %

TAUBnCDRm (slave) = 0000_H

– Duty cycle = 100 %

TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

(3) Block diagram and general timing diagram

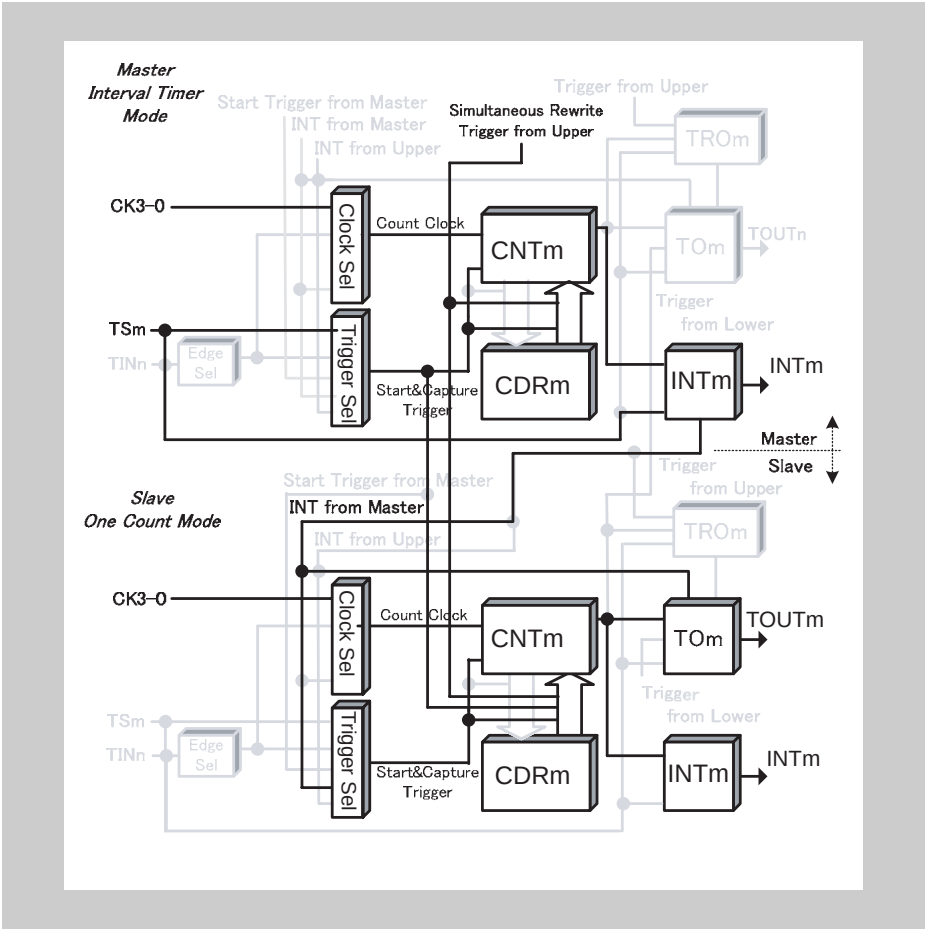


Figure 16-75 Block diagram for PWM output function

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUBnTOL.TOLm = 0)

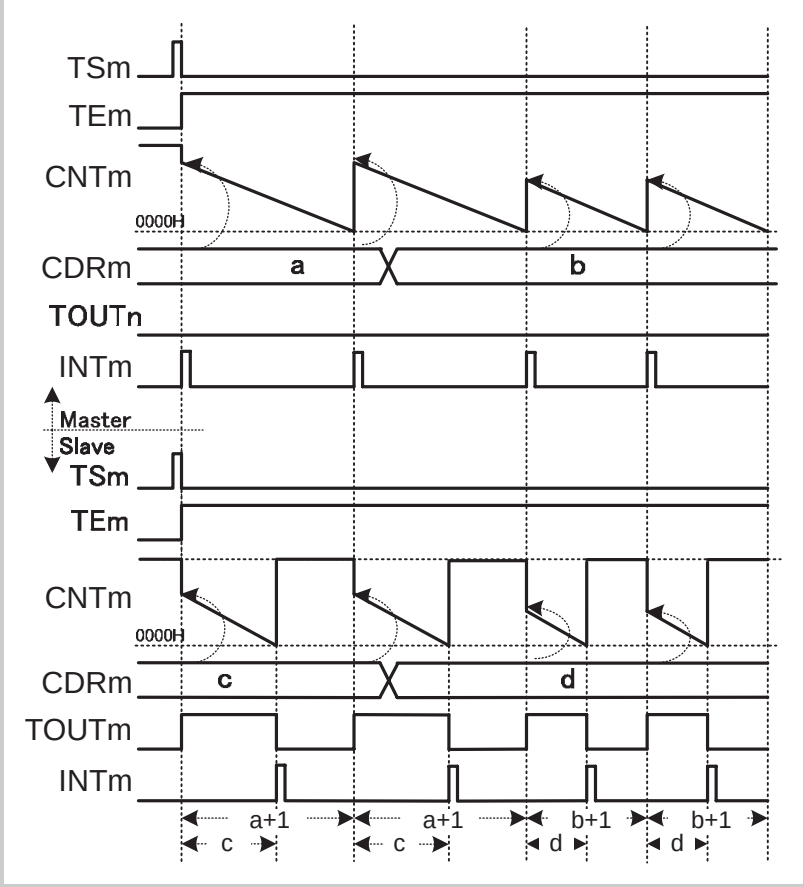


Figure 16-76 General timing diagram for PWM output function

Note The interval between the slave channel starting to count and an interrupt being generated is the value of corresponding TAUBnCDRm, whereas for the master channel the interval is the corresponding TAUBnCDRm + 1.

(4) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-75 TAUBnCMORm settings for the master channel of the PWM output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval timer mode
MD0	1: Generates INTTAUBnIm at operation start

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-76 TAUBnCMURm settings for the master channel of the PWM output function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-77 Simultaneous rewrite settings for the master channel of the PWM output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel(s)

(a) TAUBnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-78 TAUBnCMORm settings for the slave channel of the PWM output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUBnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One-count mode
MD0	1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-79 TAUBnCMURm settings for the slave channel of the PWM output function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel(s)

Table 16-80 Control bit settings for independent channel output mode 1

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	1: Synchronous channel operation
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-81 Simultaneous rewrite settings for the slave channel of the PWM output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for PWM output function

Table 16-82 Operating procedure for PWM output function

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 1074*. Slave channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(5) "Register settings for the slave channel(s)" on page 1076*. Set the values of the TAUBnCDRm registers of all channels.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUBnIm is generated on the master channel and TAUBnTTOUTm (slave) is set.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	TAUBnCNTm of the master channel loads TAUBnCDRm and counts down. When the counter reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation • TAUBnCNTm (slave) reloads the TAUBnCDRm value and counts down • TAUBnTTOUTm (slave) is set When TAUBnCNTm (slave) reaches 0000_H: • INTTAUBnIm (slave) is generated • TAUBnTTOUTm (slave) is reset
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0 %

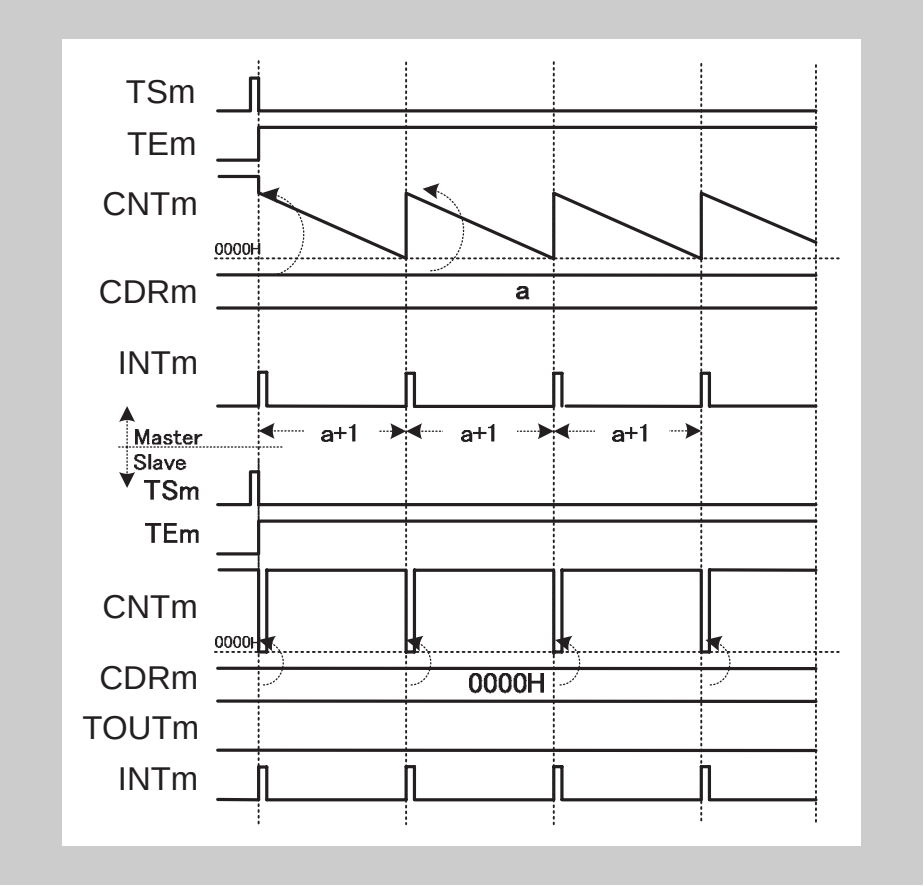


Figure 16-77 TAUBnCDRm (slave) = 0000_H, positive logic (TAUBnTOL.TOLm (slave) = 0)

- Every time the master channel generates an interrupt (INTTAUBnIm), 0000_H is written to TAUBnCNTm (slave). Therefore, TAUBnCNTm (slave) cannot start to count and TAUBnTTOUTm remains at not active state.
- TAUBnCNTm (slave) generates an interrupt every time the value of TAUBnCDRm is reloaded. The slave and the master channel generate interrupts in the same cycle.

(b) Duty cycle = 100 %

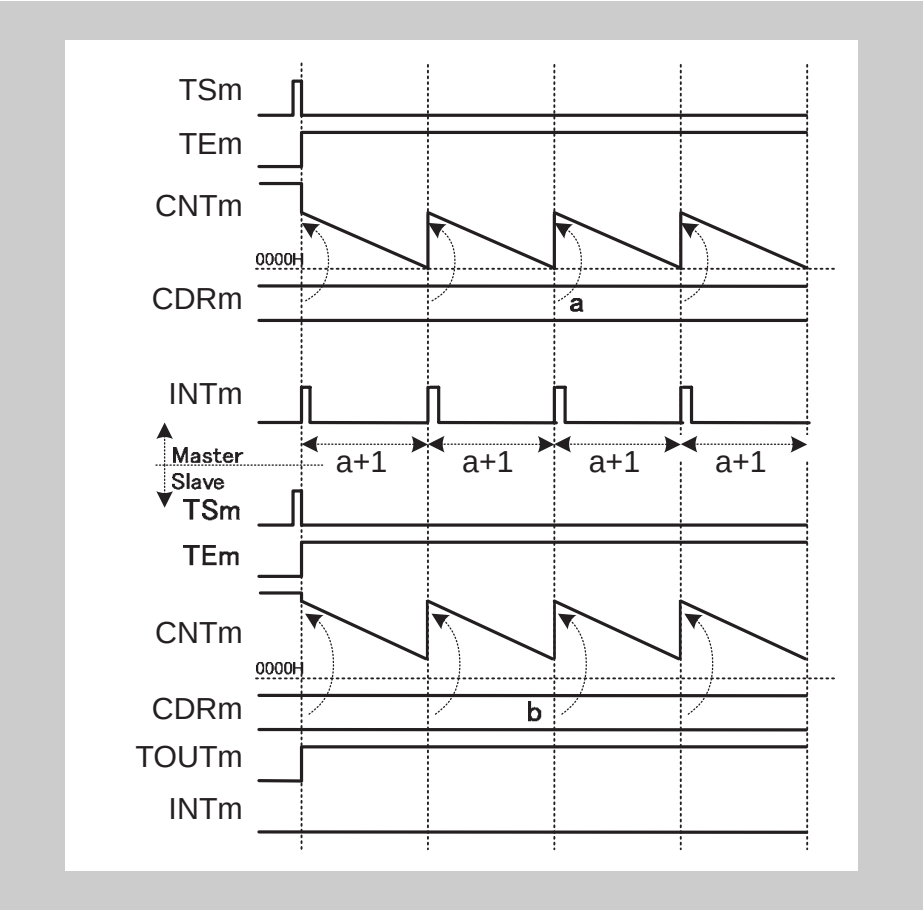


Figure 16-78 TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1, positive logic (TAUBnTOL.TOLm (slave) = 0)

- If the value TAUBnCDRm (slave) is higher than the value TAUBnCDRm (master), the counter of the slave channel cannot reach 0000_H and cannot generate interrupts. The TAUBnTTOUTm remains at active state.

(c) Stop and restart operation

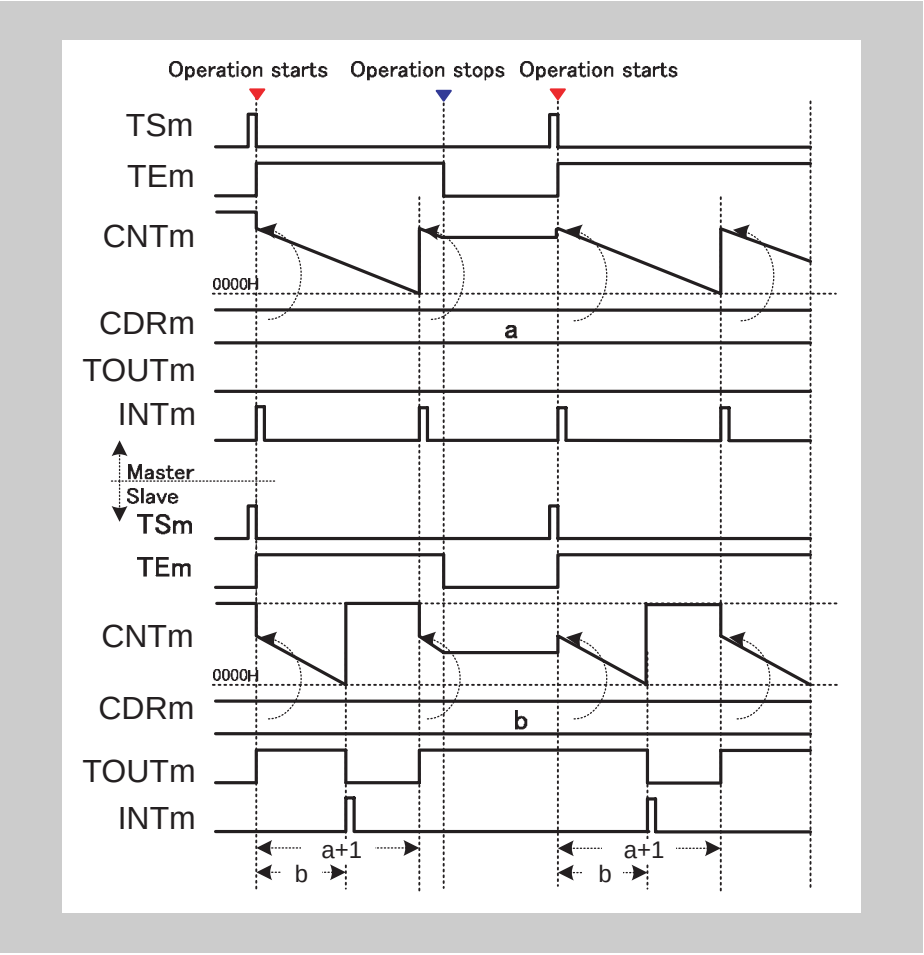


Figure 16-79 Stop and restart operation, positive logic (TAUBnTOL.TOLm (slave) = 0)

- The counter can be stopped by setting TAUBnTT.TTm of the master and slave channel(s) to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm and TAUBnTTOUTm of all channels stop and the current values are retained. No interrupts are generated.
- The counter can be restarted by setting TAUBnTS.TSm of master and slave channel(s) to 1. TAUBnCNTm of master and slave channel reload the current values of TAUBnCDRm and start to count down from these values.

16.19.2 Delay pulse output function

(1) Overview

Summary

This function outputs two signals. The reference signal has a defined pulse width and pulse cycle specified using the master channel and slave channel 1. Slave channels 2 and 3 output the reference signal with a specified delay. The delay signal is identical to the reference signal, but delayed by amount specified in slave channel 2.

The signal values are specified in the following way:

- The pulse cycle is specified using the master channel.
- The duty cycle of the reference signal is specified using slave channel 1. The duty cycle of the delay signal is specified using slave channel 3.

  The values of TAUBnCDRm of these both channels have to be identical.
- The delay is specified in slave channel 2.

Prerequisites

- Four channels
- The operation mode of the master channel must be set to interval timer mode, see *Table 16-83 “TAUBnCMORm settings for the master channel of the delay pulse output function” on page 1086*.
- The operation mode of slave channel 1 and 2 must be set to one-count mode, see *Table 16-86 “TAUBnCMORm settings for slave channel 1 of the delay pulse output function” on page 1088*.
- The operation mode of slave channel 3 must be set to pulse one-count mode, see *Table 16-90 “TAUBnCMORm settings for slave channel 2 of the delay pulse output function” on page 1090*.
- TAUBnTTOUTm is not used for the master channel and slave channel 2.
- The channel output mode of slave channel 1 must be set to synchronous channel output mode 1 (see *16.8 “Channel Output Modes” on page 962*).
- The channel output mode of slave channel 3 must be set to independent channel output mode 1 (see *16.8 “Channel Output Modes” on page 962*).

Description

The counters of the channel group are started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm, enabling count operation.

- Master channel:

  The current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value. INTTAUBnIm is generated on the master channel.
  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed and INTTAUBnIm is generated. The counter reloads the TAUBnCDRm value and counts down.
- Slave channels 1 and 2:

  When slave channels 1 and 2 detect an interrupt from the master channel, they start to count down from the current value of TAUBnCDRm. The TAUBnTTOUTm signal (slave 1) is set.

  – Slave channel 1:

    When the counter of slave channel 1 reaches 0000_H, duty time has elapsed, INTTAUBnIm is generated and the TAUBnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUBnIm of the master channel.

– Slave channel 2:

When the counter of slave channel 2 reaches 0000_H, delay time has elapsed and INTTAUBnIm is generated. The counter returns to $FFFF_H$ and awaits the next INTTAUBnIm of the master channel.

INTTAUBnIm (slave 2) triggers the counter of slave channel 3

- Slave channel 3:

When slave channel 3 detects an interrupt from slave channel 2, it starts to count down from the current value of TAUBnCDRm. INTTAUBnIm is generated and the TAUBnTTOUTm signal (slave 3) toggles.
When the counter of slave channel 3 reaches 0000_H, duty time has elapsed, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles again.
The output from slave channel 3 is the delayed PWM pulse

The counter can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channels, which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUBnTS.TSm to 1.

Conditions Simultaneous rewrite can be used with this function. See *16.7 “Simultaneous Rewrite” on page 952*.

Equations Pulse cycle = (TAUBnCDRm (master) + 1) × count clock cycle

Duty cycle = (TAUBnCDRm (slave 1 and 3)) × count clock cycle

Delay = (TAUBnCDRm (slave 2) + 1) × count clock cycle

(2) Block diagram and general timing diagram

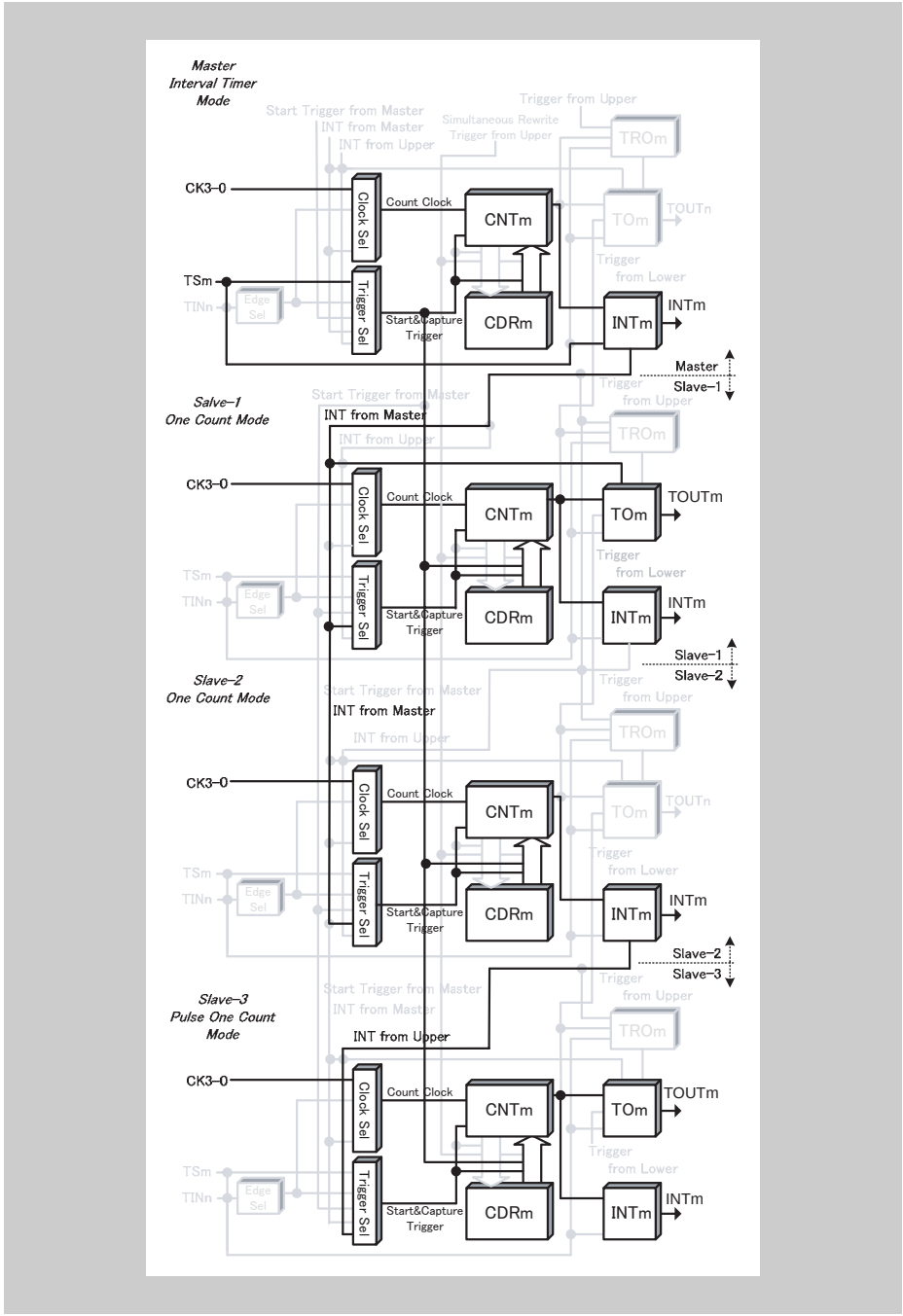


Figure 16-80 Block diagram for delay pulse output function

The following settings apply to the general timing diagram:

- All channels
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)

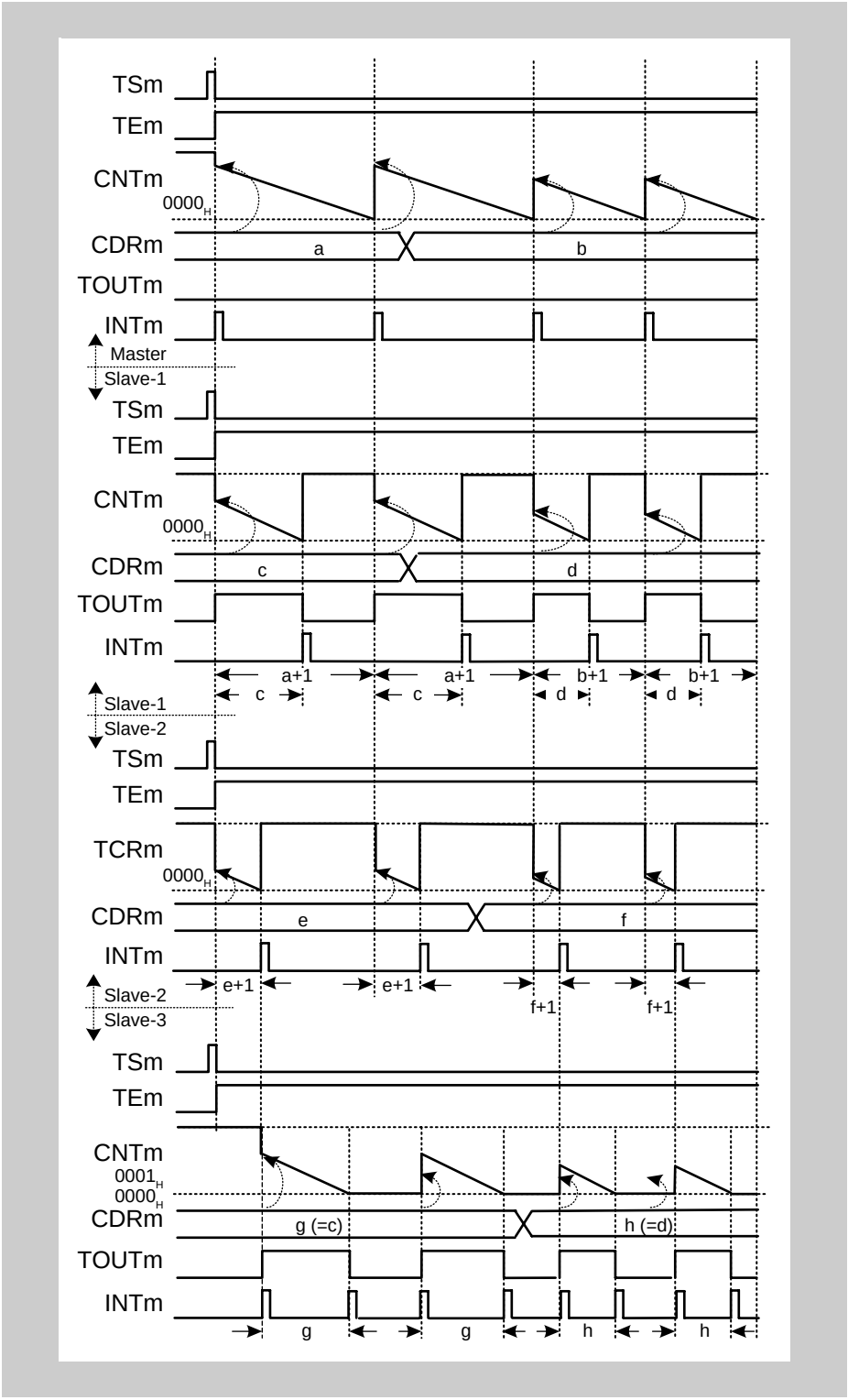


Figure 16-81 General timing diagram for delay pulse output function

(3) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-83 TAUBnCMORm settings for the master channel of the delay pulse output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval timer mode
MD0	1: Generates INTTAUBnIm at operation start

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-84 TAUBnCMURm settings for the master channel of the delay pulse output function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

The channel output mode is not used by the master channel of this function. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-85 Simultaneous rewrite settings for the master channel of the delay pulse output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(4) Register settings for slave channel 1

(a) TAUBnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-86 TAUBnCMORm settings for slave channel 1 of the delay pulse output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: TAUBnTTOUTm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One-count mode
MD0	1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start (and enables start trigger during counting).

(b) TAUBnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-87 TAUBnCMURm settings for slave channel 1 of the delay pulse output function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 1

Table 16-88 Control bit settings for slave channel 1 of the synchronous channel output mode 2

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	1: Synchronous channel operation
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-89 Simultaneous rewrite settings for slave channel 1 of the delay pulse output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 2

(a) TAUBnCMORm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-90 TAUBnCMORm settings for slave channel 2 of the delay pulse output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: TAUBnTTOUTm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One-count mode
MD0	1: Generates INTTAUBnIm at operation start (and enables start trigger during counting)

(b) TAUBnCMURm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-91 TAUBnCMURm settings for slave channel 2 of the delay pulse output function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 2

The channel output mode is not used by this function. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for slave channel 2

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-92 Simultaneous rewrite settings for slave channel 2 of the delay pulse output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channel 3

(a) TAUBnCMORm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-93 TAUBnCMORm settings for slave channel 3 of the delay pulse output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	101: INTTAUBnIm of the upper channel (m-1) is the start trigger, regardless of the master setting
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1010: Pulse one-count mode
MD0	1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start (and enables start trigger during counting)

(b) TAUBnCMURm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-94 TAUBnCMURm settings for slave channel 3 of the delay pulse output function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 3

Table 16-95 Control bit settings for synchronous channel output mode 2

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	0: Independent channel output
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for slave channel 3

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-96 Simultaneous rewrite settings for slave channel 3 of the delay pulse output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for delay pulse output function

Table 16-97 Operating procedure for delay pulse output function (1/2)

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(3) "Register settings for the master channel" on page 1086*. Slave channel 1: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(4) "Register settings for slave channel 1" on page 1088*. Slave channel 2: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(5) "Register settings for slave channel 2" on page 1090*. Slave channel 3: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(6) "Register settings for slave channel 3" on page 1092*. Set the values of the TAUBnCDRm registers of all channels.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)

Table 16-97 Operating procedure for delay pulse output function (2/2)

	Operation	Status of TAUBn
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the counters of the master channel and slave channels 1 and 2 start. INTTAUBnIm is generated on the master channel and TAUBnTTOUTm (slave 1) is set.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	TAUBnCNTm of the master channel and slave channels 1 and 2 load TAUBnCDRm and count down. When the counter of the master channel reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation • TAUBnCNTm (slave 1 and slave 2) reload the TAUBnCDRm value and start counting down • TAUBnTTOUTm (slave 1) is set When TAUBnCNTm (slave 1) reaches 0000_H: • INTTAUBnIm (slave 1) is generated The output from slave channel 1 acts as the reference pulse • TAUBnTTOUTm (slave 1) is reset When TAUBnCNTm (slave 2) reaches 0000_H: • INTTAUBnIm (slave 2) is generated • TAUBnTTOUTm (slave 3) toggles • TAUBnCNTm (slave 3) reloads the TAUBnCDRm value and starts counting down When TAUBnCNTm (slave 3) reaches 0000_H: • INTTAUBnIm (slave 3) is generated • TAUBnTTOUTm (slave 3) toggles The output from slave channel 3 is the delayed PWM pulse
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

(8) Specific timing diagrams

(a) Duty cycle (slave 3) = 100 %

The following values apply to the figure below:

- TAUBnCDRm (master) = $000A_H$
- TAUBnCDRm (slave 1) = $000B_H$
- TAUBnCDRm (slave 2) = 0000_H
- TAUBnCDRm (slave 3) = $000B_H$

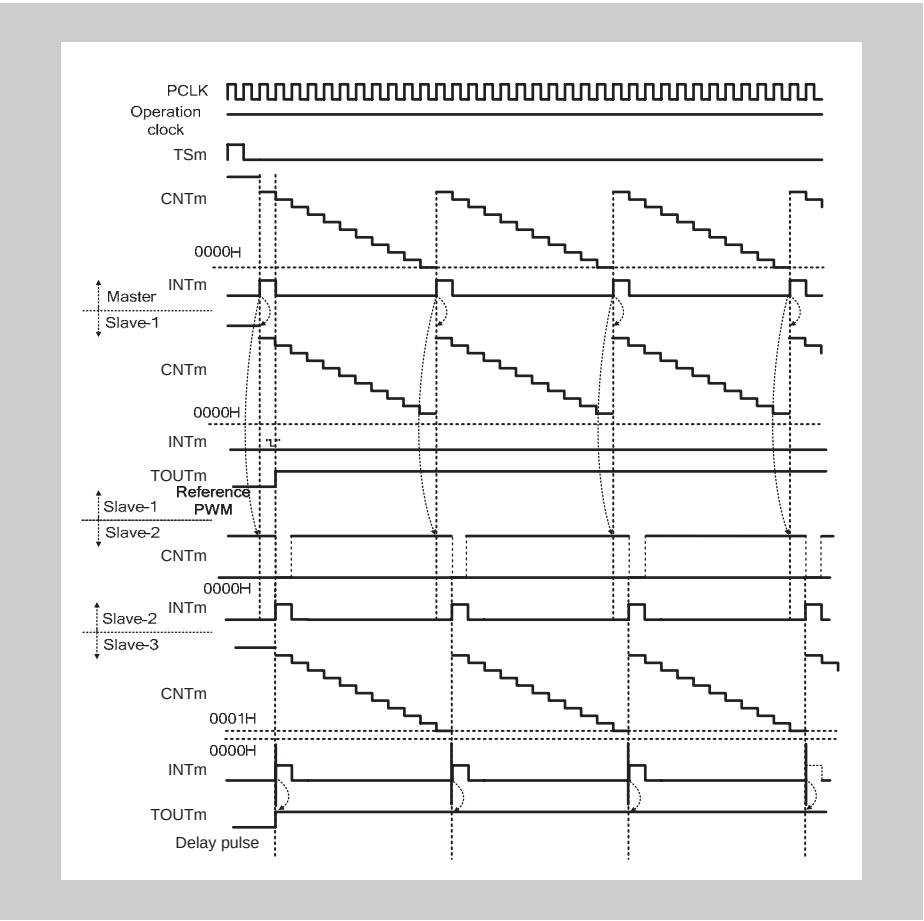


Figure 16-82 Duty cycle (slave 3) = 100 %

- If the value of TAUBnCDRm (slave 1 and 3) is higher than the value of TAUBnCDRm (master), the counter of the slave channels cannot reach 0000_H and cannot generate interrupts. TAUBnTTOUTm of channels 1 and 3 remain in the active state.

(b) TAUBnTTOUTm (slave 1) = TAUBnTTOUTm (slave 3)

The following values apply to the figure below:

- TAUBnCDRm (master) = $000A_H$
- TAUBnCDRm (slave 1) = 0005_H
- TAUBnCDRm (slave 2) = 0000_H
- TAUBnCDRm (slave 3) = 0005_H

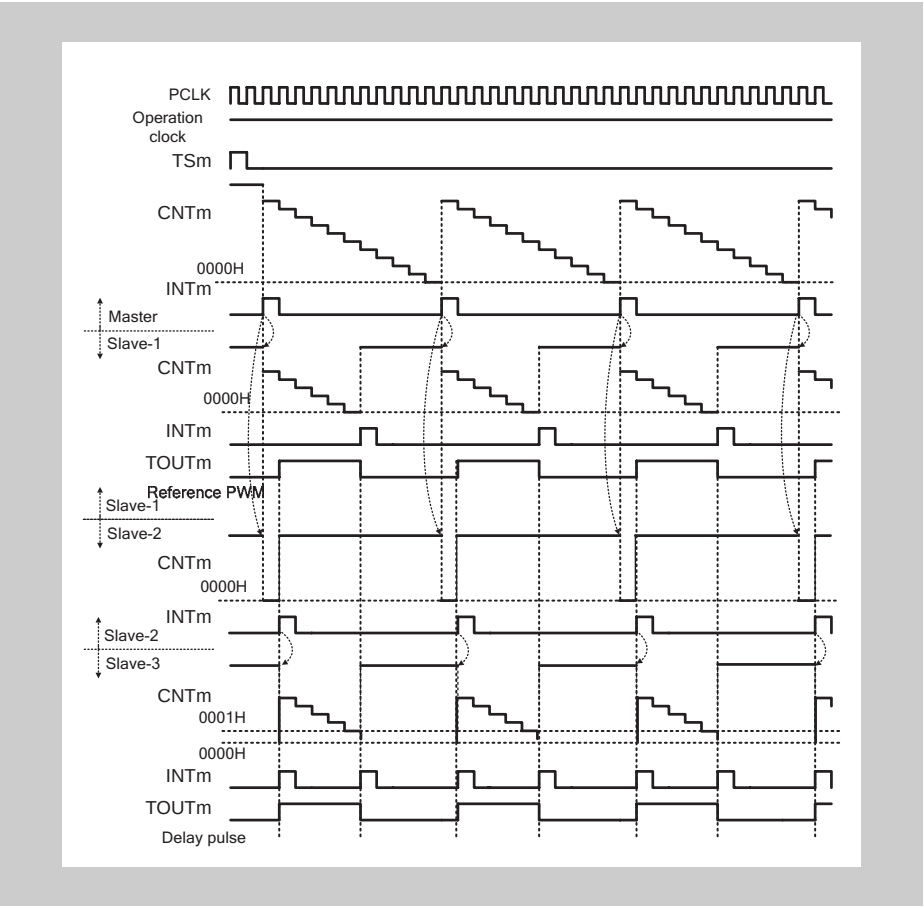


Figure 16-83 TAUBnTTOUTm (slave 1) = TAUBnTTOUTm (slave 3)

- If TAUBnCDRm (slave 2) = 0000_H, the counter of slave channel 3 starts counting one count clock later than the counter of slave channel 1. The reference pulse and the delay pulse are output with a delay of one clock count.

16.19.3 AD conversion trigger output function type 1

(1) Overview

Summary This function is identical to *16.19.1 "PWM output function" on page 1071* except that TAUBnTTOUTm is not output.

This is achieved by setting the channel output mode of the slave to independent channel output mode controlled by software.

(2) Block diagram and general timing diagram

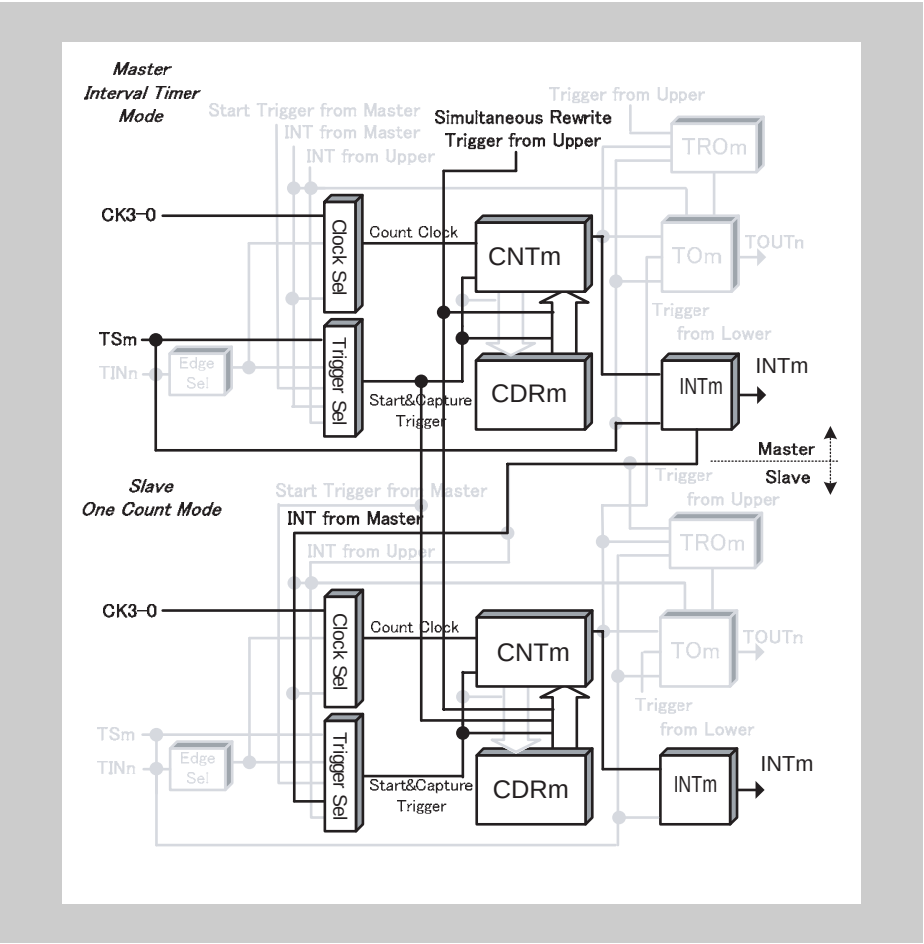


Figure 16-84 Block diagram for AD conversion trigger output function type 1

(3) General timing diagram

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUBnTOL.TOLm = 0)

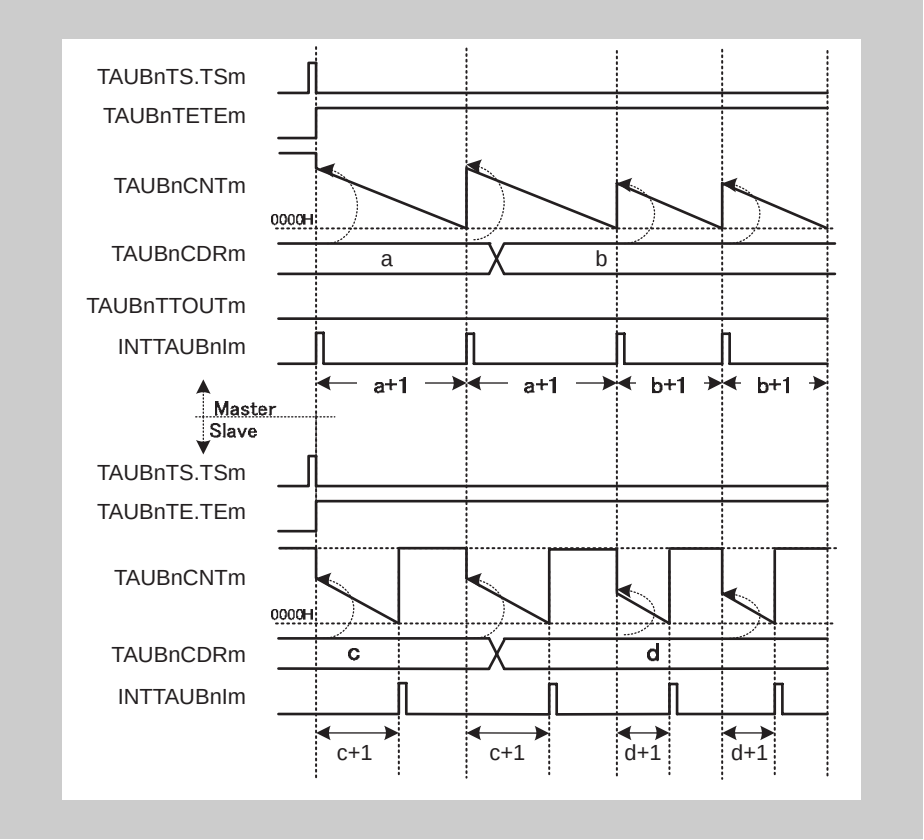


Figure 16-85 General timing diagram for AD conversion trigger output function type 1

16.20 Synchronous PWM Signal Functions Triggered by an External Signal

This chapter describes functions that generate PWM signals and which are triggered by an external signal.

- *16.20.1 “One-shot pulse output function”*

16.20.1 One-shot pulse output function

(1) Overview

Summary This function outputs a signal pulse with a defined pulse width and a specific delay time compared to an external input signal pulse by using a master and a slave channel. The delay time is specified using the master channel. The pulse width is specified using the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to one-count mode, see *Table 16-98 "TAUBnCMORm settings for the master channel of the one-shot pulse output function" on page 1104*.
- The operation mode of the slave channel must be set to Pulse one-count mode, see *Table 16-101 "TAUBnCMORm settings for the slave channel of the one-shot pulse output function" on page 1106*.
- TAUBnTTOUTm is not used for the master channel of this function.
- The channel output mode of the slave channel must be set to synchronous channel output mode 2 (see *16.8 "Channel Output Modes" on page 962*).
- TAUBnTTINm (master) has to be detected while TAUBnCNTm (master) and TAUBnCNTm (slave) await a trigger. Furthermore, the slave is only triggered by an interrupt from the master channel and not by TAUBnTTINm.

Description The counters are enabled by setting the channel trigger bits (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm, enabling count operation.

- Master channel:

  When the next valid TAUBnTTINm input edge is detected, the current value of TAUBnCDRm is written to TAUBnCNTm. The counter starts to count down from this value. If TAUBnCMORm.MD0 = 0, a trigger (TAUBnTTINm) which is detected within the delay time is ignored.
  When the counter of the master channel reaches 0000_H, INTTAUBnIm is generated. The counter returns to $FFFF_H$ and awaits the next valid TAUBnTTINm input edge.

- Slave channel

  The INTTAUBnIm of the master channel triggers the counter of the slave channel. The current value of TAUBnCDRm (slave) is written to TAUBnCNTm (slave) and the counter starts to count down from this value. An interrupt is generated and the TAUBnTTOUTm signal is set.
  When the counter reaches 0001_H, INTTAUBnIm is generated and the TAUBnTTOUTm signal is reset. The counter remains at 0001_H and awaits the next INTTAUBnIm of the master channel.

The counter can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channel, which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channel stop but retain their values. The counters can be restarted by setting TAUBnTS.TSm to 1.

The counter of the master channel can be restarted without stopping it first (forced restart) by setting TAUBnTS.TSm to 1 during operation.

Notes

1. If a forced restart of the slave channel is executed during operation, the width of the output signal does not correspond to the value of TAUBnCDRm (slave).
2. The input TAUBnTTINm is sampled at the frequency of the operating clock, specified by TAUBnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUBnTTOUTm has an error of ± 1 operation clock cycle.

Conditions

- If TAUBnCMORn.MD0 of the master channel is set to 0, during counting detected TAUBnTTINm input edges are ignored.
- Simultaneous rewrite can be used with this function. See *16.7 "Simultaneous Rewrite" on page 952*.

Equations

Delay to input pulse = (TAUBnCDRm (master) + 1) × count clock cycle

Pulse width = (TAUBnCDRm (slave)) × count clock cycle

(2) Block diagram and general timing diagram

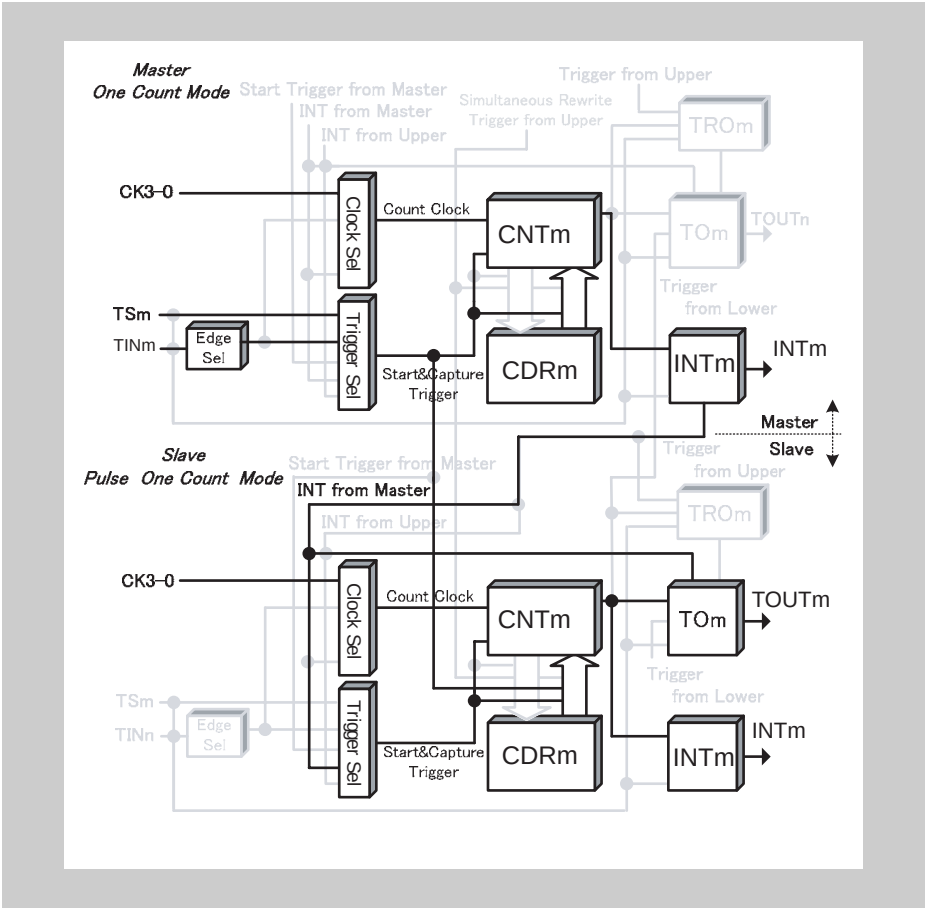


Figure 16-86 Block diagram for one-shot pulse output function

The following settings apply to the general timing diagram:

- Start trigger detection disabled during counting (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

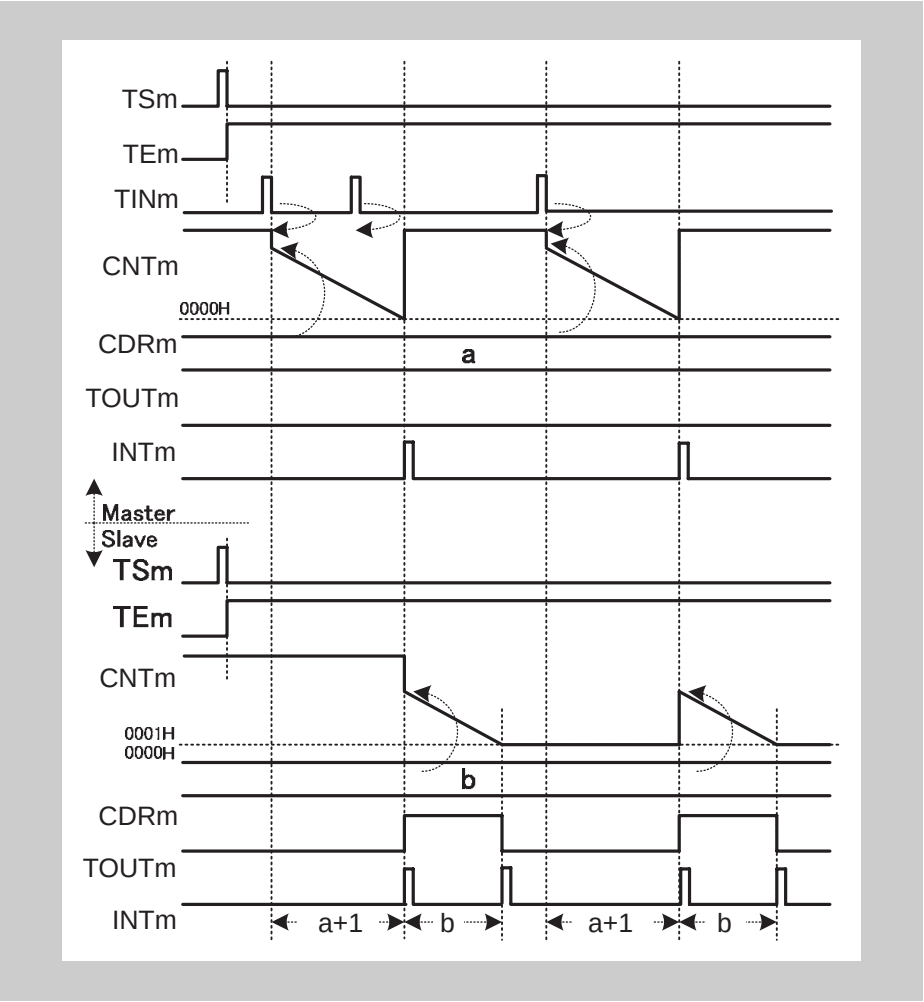


Figure 16-87 General timing diagram for one-shot pulse output function

(3) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-98 TAUBnCMORm settings for the master channel of the one-shot pulse output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	001: Valid TAUBnTTINm input edge signal is used as the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One-count mode
MD0	0: Disables start trigger detection during counting 1: Enables start trigger detection during counting The value of the MD0 bit of the master and slave channel must be identical.

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-99 TAUBnCMURm settings for the master channel of the one-shot pulse output function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in independent channel output mode controlled by software.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-100 Simultaneous rewrite settings for the master channel of the one-shot pulse output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(4) Register settings for the slave channel

(a) TAUBnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-101 TAUBnCMORm settings for the slave channel of the one-shot pulse output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: TAUBnTTOUTm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1010: Pulse one-count mode
MD0	0: Disables start trigger detection during counting 1: Enables start trigger detection during counting The value of the MD0 bit of the master and slave channel must be identical.

(b) TAUBnCMURm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-102 TAUBnCMURm settings for the slave channel of the one-shot pulse output function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel

Table 16-103 Control bit settings for synchronous channel output mode 2

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	0: Independent channel output
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-104 Simultaneous rewrite settings for the slave channel of the one-shot pulse output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Operating procedure for one-shot pulse output function

Table 16-105 Operating procedure for one-shot pulse output function

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(3) "Register settings for the master channel" on page 1104*. Slave channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(4) "Register settings for the slave channel" on page 1106*. Set the values of the TAUBnCDRm registers of all channels.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the master channel awaits a TAUBnTTINm input. INTTAUBnIm is generated on the master channel.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	When a valid TAUBnTTINm input edge is detected, TAUBnCNTm of the master channel loads TAUBnCDRm and counts down. When the counter reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation • TAUBnCNTm (slave) reloads the TAUBnCDRm value and starts to count down • INTTAUBnIm (slave) is generated • TAUBnTTOUTm (slave) is set When TAUBnCNTm (slave) reaches 0001_H: • INTTAUBnIm (slave) is generated • TAUBnTTOUTm (slave) is reset If a TAUBnTTINm input is detected on the master channel while the counter is counting, the input is ignored when TAUBnCMORm.MD0 = 0.
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

(6) Specific timing diagrams

(a) TAUBnCDRm (master) = 0000_H

The following settings apply to this diagram:

- Start trigger detection disabled during counting (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

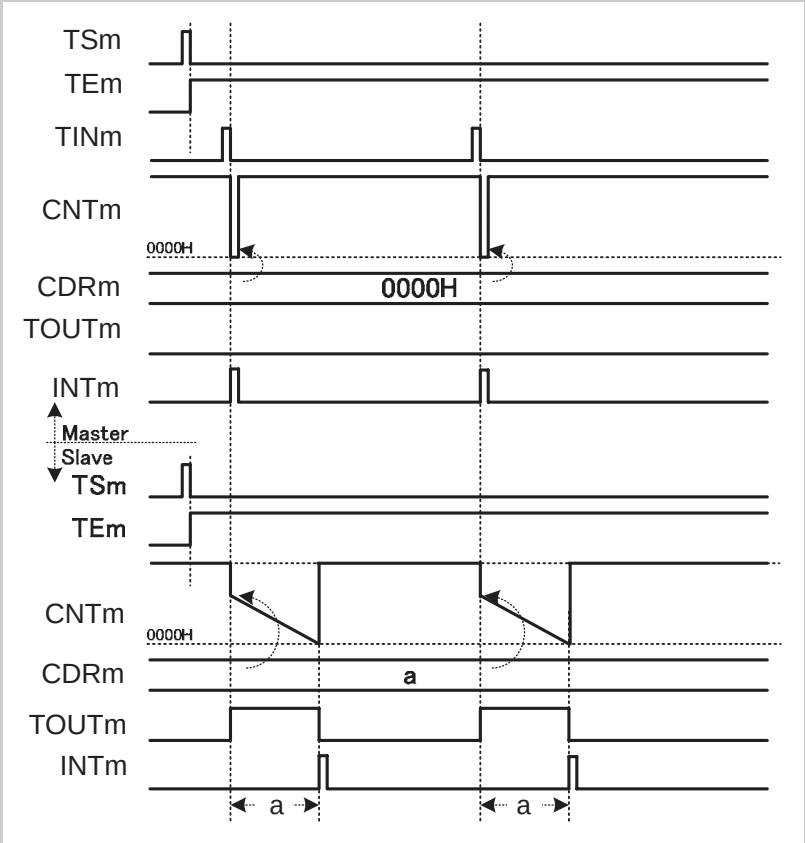


Figure 16-88 TAUBnCDRm (master) = 0000_H

- When a valid TAUBnTTINm input edge is detected, the value 0000_H is written to TAUBnCNTm (master). The counter is set to 0000_H for one count and returns to $FFFF_H$.

  Thus the slave channel starts to count down one count clock later to TAUBnTTINm (master).

(b) TAUBnCDRm (slave) = 0000_H

The following settings apply to this diagram:

- Start trigger detection disabled during counting (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

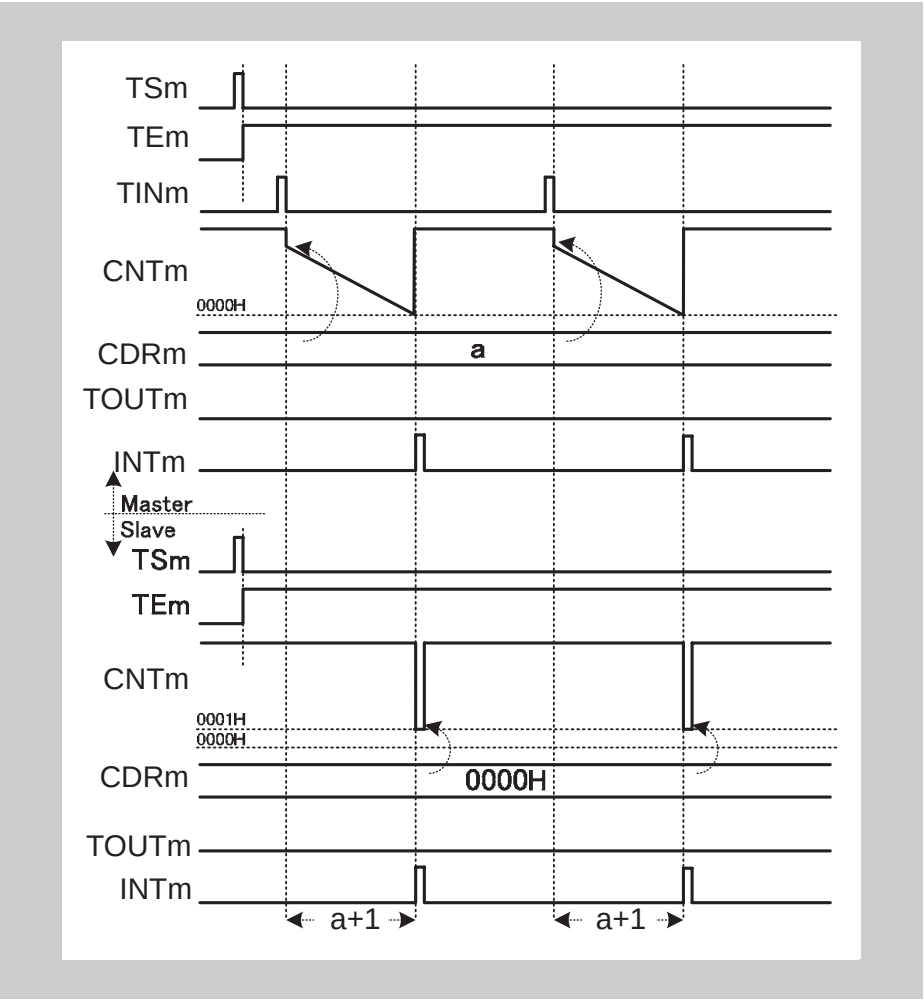


Figure 16-89 TAUBnCDRm (slave) = 0000_H

- The counter of the slave channel reloads the value 0000_H and returns to returns to $FFFF_H$ one clock count later.

  TAUBnTTOUTm remains at not active state, because the pulse width is zero.

(c) TAUBnCMORm.MD0 = 1

The following settings apply to this diagram:

- Start trigger detection enabled during counting (TAUBnCMORm.MD0 = 1)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

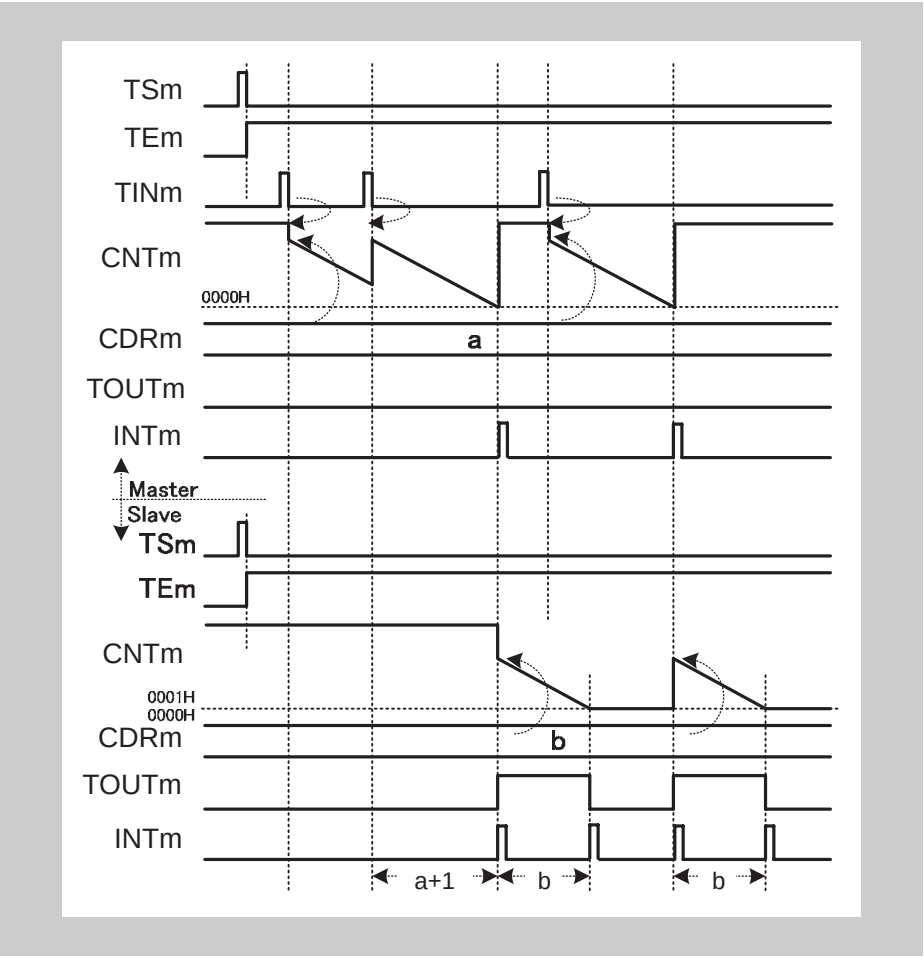


Figure 16-90 TAUBnCMORm.MD0 = 1

- If a valid TAUBnTTINm input edge is detected while the counter of the master channel counts down, TAUBnCNTm reloads the value of TAUBnCDRm. The counter restarts to count down.

  This means the delay is extended by the value of TAUBnCNTm at the time a valid TAUBnTTINm input edge is detected.

(d) Restarting the master channel while the slave channel is counting

The following settings apply to this diagram:

- Start trigger detection disabled during counting (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

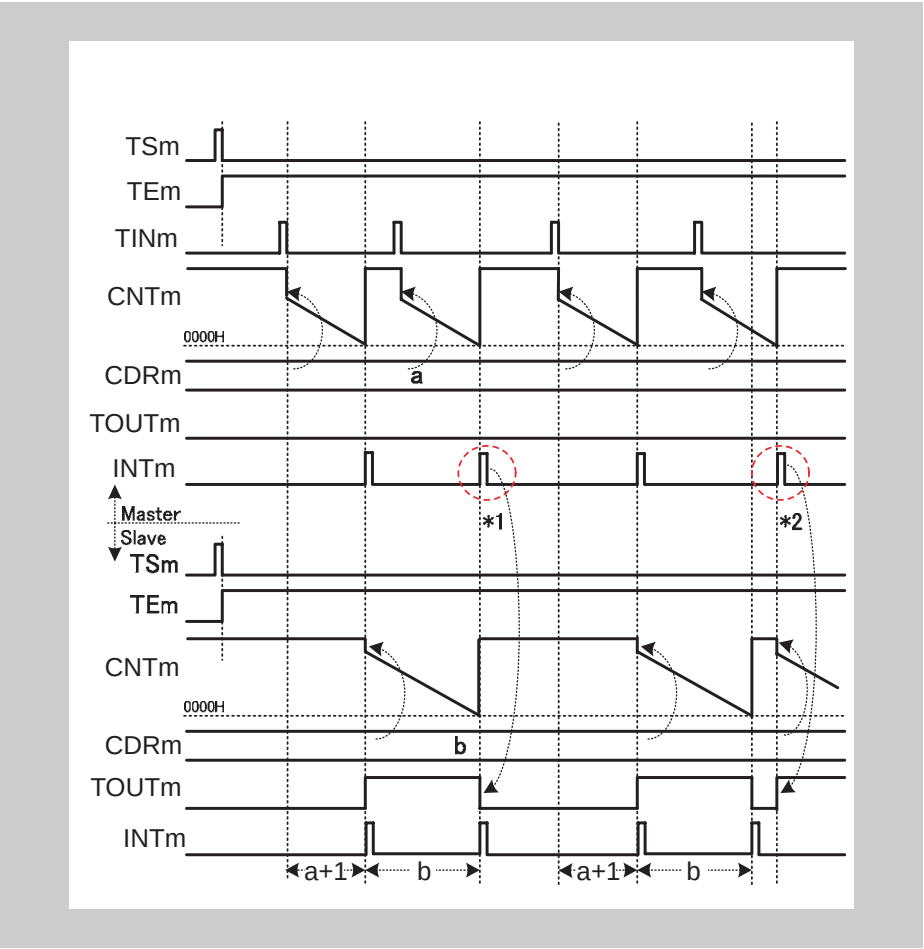


Figure 16-91 Interval of TAUBnTTINm ≤ delay time + pulse width +1

- If the master channel generates an interrupt before the counter of the slave channel has reached 0001_H or exactly when 0001_H is reached (*1), the interrupt (master) is ignored.
- If an interrupt of the master channel occurs when the counter of the slave channel awaits the next trigger, the value of TAUBnCDRm (slave) is reloaded. An interrupt is generated and TAUBnTTOUTm toggles. If TAUBnCNTm (master) has started to count down while the TAUBnCNTm (slave) is still counting (*2), TAUBnTTOUTm is not output with the expected delay time.
- To generate the correct one-shot pulse, the start trigger for the master channel must be detected while the master and slave channels are waiting for the start trigger, and not while they are counting.

16.21 Synchronous Triangle PWM Functions

This chapter describes functions that generate a triangle PWM output.

- *16.21.1 “Triangle PWM output function”*
- *16.21.2 “Triangle PWM output function with dead time”*
- *16.21.3 “AD conversion trigger output function type 2”*

16.21.1 Triangle PWM output function

(1) Overview

Summary This function generates multiple triangle PWM outputs by using a master and one or more slave channels. It enables the pulse cycle (frequency) and the duty cycle of TAUBnTTOUTm to be set using the master and slave channel(s) respectively.

The slave channel generates a carrier cycle from two pulse cycles. The first pulse of the master channel controls the down status and the second pulse controls the up status of the slaves counter.

Counting up and down TAUBnCNTm (slave) means that signal duration of TAUBnTTOUTm (slave) is double that of the difference between TAUBnCDRm (master) +1 and TAUBnCDRm (slave).

Prerequisites

- Two channels
- The operation mode of the master channel must be set to interval timer mode, see *Table 16-106 “TAUBnCMORm settings for the master channel of the triangle PWM output function” on page 1118*.
- The operation mode of the slave channel(s) must be set to up/down count mode, see *Table 16-110 “TAUBnCMORm settings for the slave channel of the triangle PWM output function” on page 1120*.
- The channel output mode of the master channel must be set to independent channel output mode 1 (see *16.8 “Channel Output Modes” on page 962*).
- The channel output mode of the slave channel(s) must be set to synchronous channel output mode 2 (see *16.8 “Channel Output Modes” on page 962*).
- The following settings establish TAUBnTTOUTm at high level for the down status of the carrier cycle.
  - If the TAUBnCMORm.MD0 (master) bit is set to 0,TAUBnTO.TOm must be set to 1 while TAUBnTOE.TOEm is 0.
  - If the TAUBnCMORm.MD0 (master) bit is set to 1, TAUBnTO.TOm must be set to 0 while TAUBnTOE.TOEm is 0.

Description The counters are started by setting the channel trigger bit (TAUBnTS.TSm) to 1 for every channel. This in turn sets TAUBnTE.TEm, enabling count operation. The current values of TAUBnCDRm (master and slave) are written to TAUBnCNTm (master and slave) and the counters start to count down from these values. Depending on the setting of the master channel TAUBnCMORm.MD0 bit an interrupt is generated and TAUBnTTOUTm signal of the master toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles. TAUBnCNTm then reloads the TAUBnCDRm value and counts down.

- Slave channel:

  The INTTAUBnIm of the master channel triggers the counter of the slave channel:

  – If the slave counter currently counts down, it changes count direction.

  – If the slave counter currently counts up, the value of TAUBnCDRm is reloaded and the counter counts down.

  When the counter of the slave channel reaches 0001_H while counting up or down, INTTAUBnIm is generated and the TAUBnTTOUTm (slave) signal toggles:

  – It is set in the count-down status

  – It is reset in the count-up status

  The counter continues to count down or up and awaits the next INTTAUBnIm of the master channel.

TAUBnTTOUTm can be switched between positive and negative phase setting TAUBnTOL.TOLm during operation.

The counters can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channel(s) stop but retain their values.

Note If a forced restart is executed during operation, TAUBnTTOUTm is not output as a triangle PWM signal.

Conditions Simultaneous rewrite can be used with this function. See *16.7 “Simultaneous Rewrite” on page 952*.

(2) Equations

Pulse cycle = (TAUBnCDRm (master) + 1) x count clock cycle

Carrier cycle (down/up) = (TAUBnCDRm (master) + 1) × 2 × count clock cycle

Duty cycle [%] =
[(TAUBnCDRm (master) + 1 - TAUBnCDRm (slave))/ (TAUBnCDRm (master) + 1)] x 100

– Duty cycle = 100 %

  TAUBnCDRm (slave) = 0000_H

– Duty cycle = 0 %

  TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

(3) Block diagram and general timing diagram

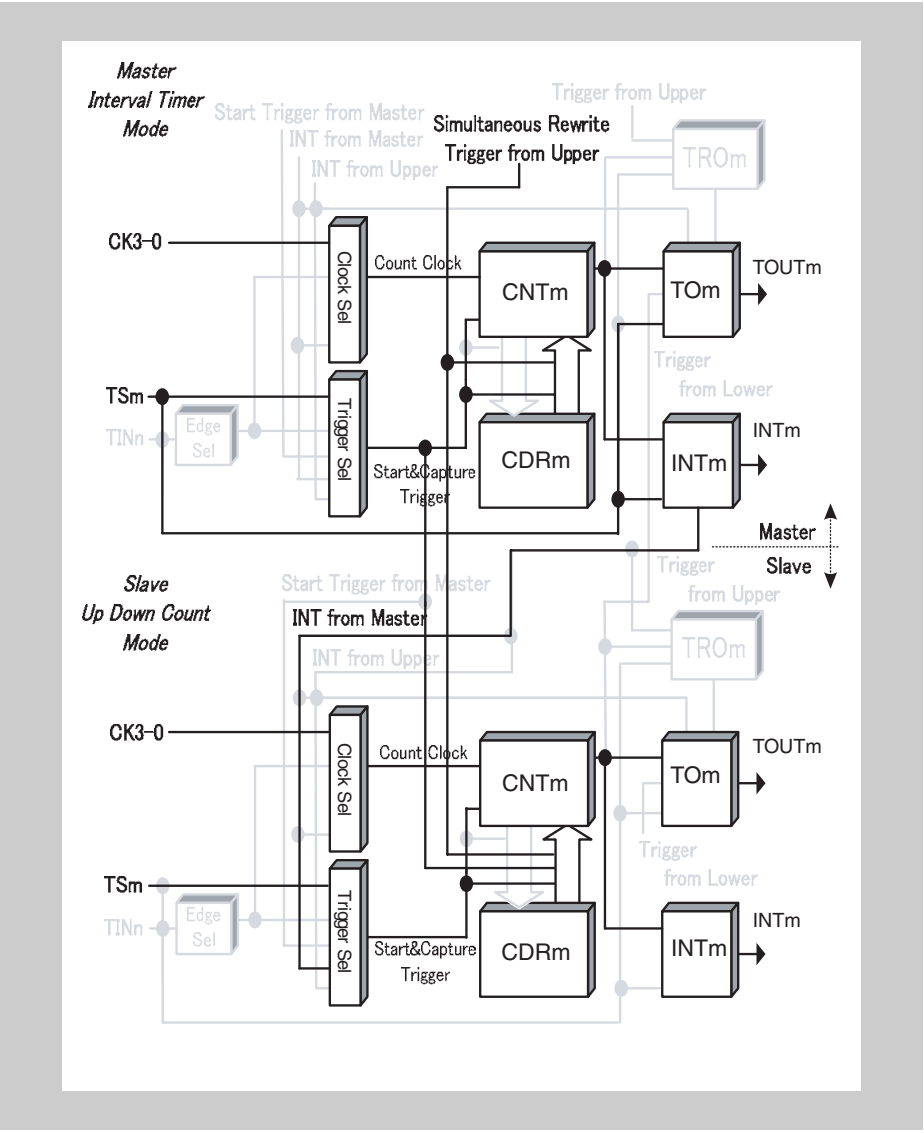


Figure 16-92 Block diagram for triangle PWM output function

The following settings apply to the general timing diagram:

- Master channel
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)

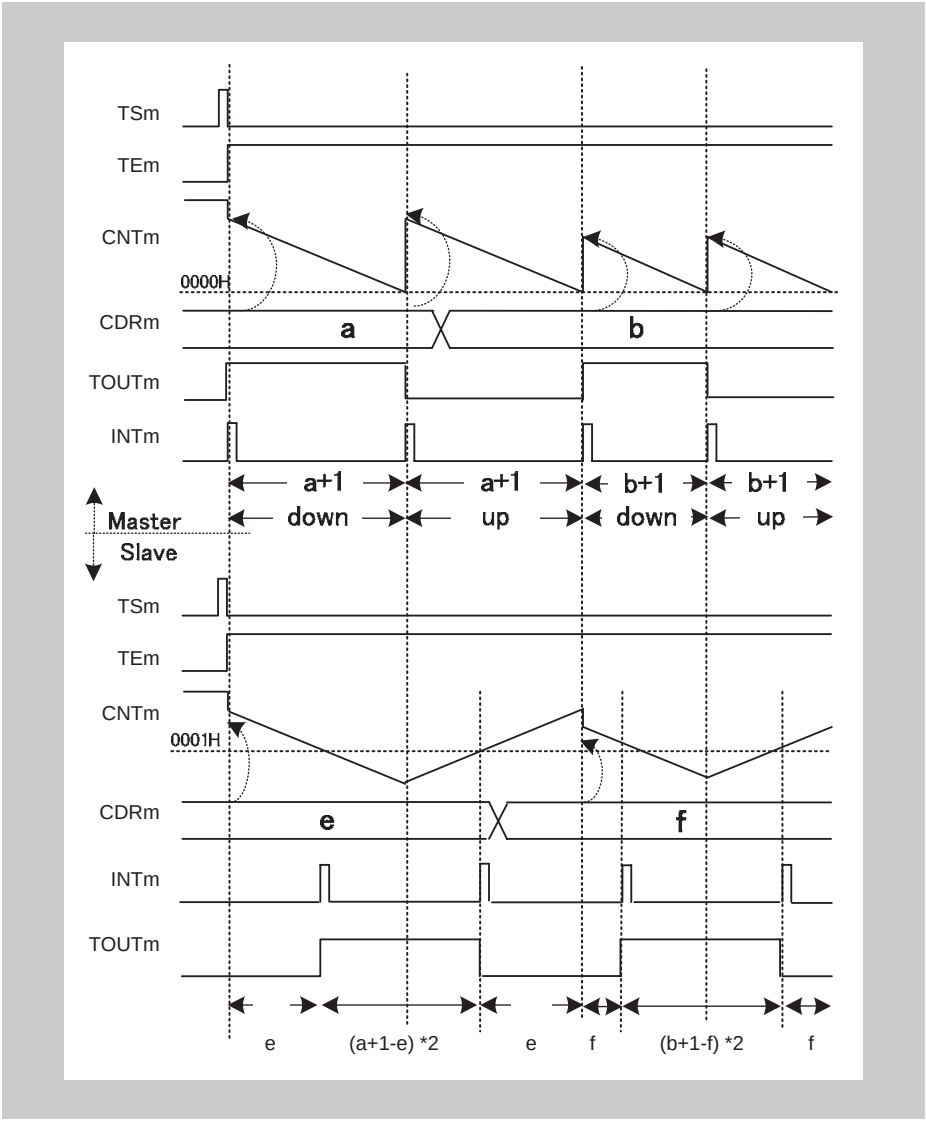


Figure 16-93 General timing diagram for triangle PWM output function

(4) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-106 TAUBnCMORm settings for the master channel of the triangle PWM output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval timer mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-107 TAUBnCMURm settings for the master channel of the triangle PWM output function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

Table 16-108 Control bit settings for independent channel output mode 1

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	0: Independent channel output
TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOLm	0: Positive logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-109 Simultaneous rewrite settings for the master channel of the triangle PWM output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel(s)

(a) TAUBnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-110 TAUBnCMORm settings for the slave channel of the triangle PWM output function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUBnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1001: Up/down count mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-111 TAUBnCMURm settings for the slave channel of the triangle PWM output function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel(s)

Table 16-112 Control bit settings for synchronous channel output mode 2

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	1: Synchronous channel operation
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-113 Simultaneous rewrite settings for the slave channel of the triangle PWM output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for triangle PWM output function

Table 16-114 Operating procedure for triangle PWM output function

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 1118*. Slave channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(5) "Register settings for the slave channel(s)" on page 1120*. Set the values of the TAUBnCDRm registers of all channels.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUBnIm is generated on the master channel, depending on TAUBnCMORm.MD0.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	TAUBnCNTm of the master and slave channel loads TAUBnCDRm and counts down. When the counter of the master channel reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnTTOUTm (master) toggles • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation. • TAUBnCNTm (slave) reloads the TAUBnCDRm value or counts in the reverse direction. When TAUBnCNTm of the slave = 0001_H: • INTTAUBnIm (slave) is generated • TAUBnTTOUTm (slave) is set (in count-down status) or reset (in count-up status)
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm is initialized to the value set by TAUBnTO.TOm.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0 %

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)
  - TAUBnCDRm = a = 5_H
- Slave channel:
  - TAUBnCDRm = 6_H

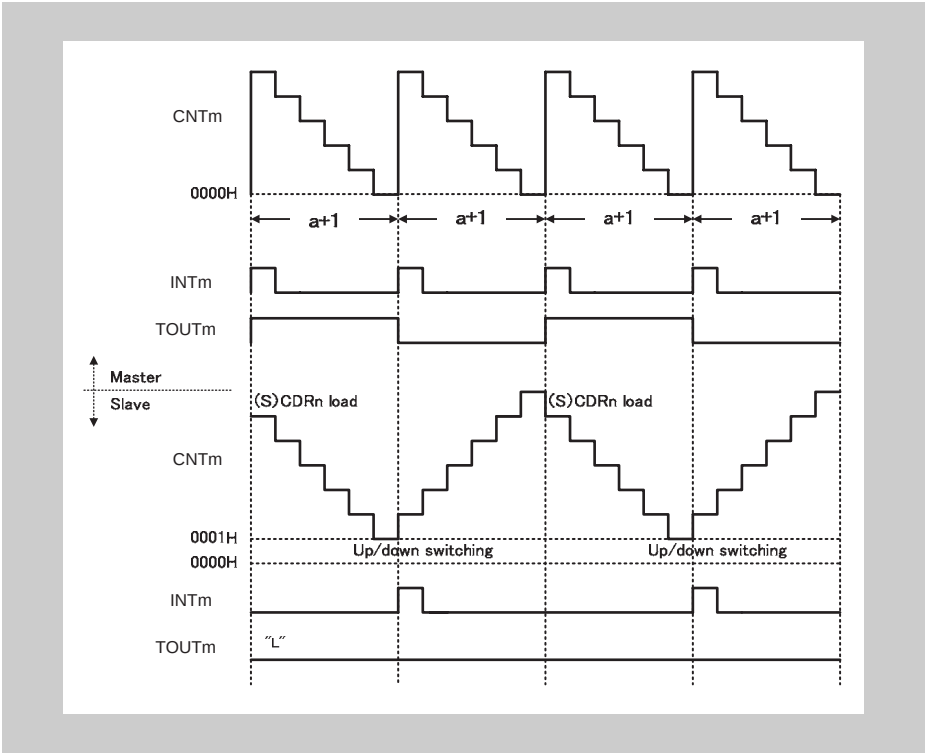


Figure 16-94 TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

- If TAUBnCDRm (slave) ≥ TAUBnCDRm (master) the counter of slave channel cannot reach 0001_H during *counting down*. The set signal is never detected, so TAUBnTTOUTm remains at low state.

(b) Duty cycle = 100 %

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)
  - TAUBnCDRm = a = 5_H
- Slave channel:
  - TAUBnCDRm = 0_H

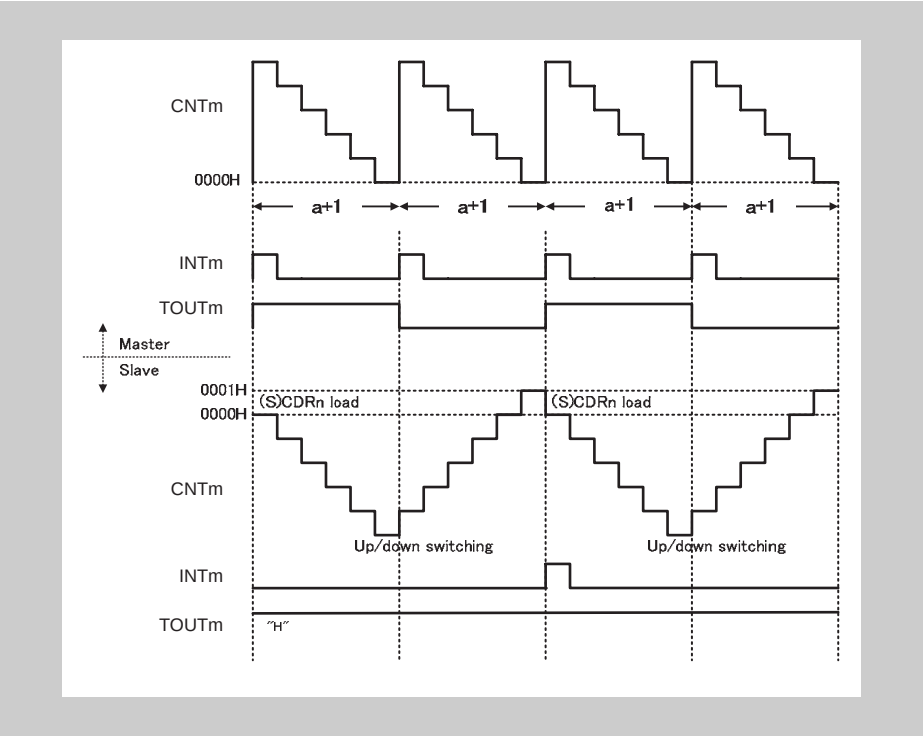


Figure 16-95 TAUBnCDRm (slave) = 0000_H

- If TAUBnCDRm (slave) =0000_H the counter of slave channel cannot reach 0001_H during *counting up*. The reset signal is never detected, so TAUBnTTOUTm remains at high state.

16.21.2 Triangle PWM output function with dead time

(1) Overview

Summary This function generates multiple triangle PWM outputs with a defined dead time by using a master and two or more slave channels. The resulting PWM signals are output via TAUBnTTOUTm of slave channels 2 and 3. It enables the pulse cycle (frequency) and the duty cycle of TAUBnTTOUTm to be set using the master and slave channel(s) respectively.

Slave channel 2 generates a carrier cycle from two pulse cycles from the master channel. The first pulse controls the down status and the second pulse controls the up status of the slaves counter.

Counting up and down TAUBnCNTm (slave 2) means that signal duration of TAUBnTTOUTm (slave) is double that of the difference between TAUBnCDRm (master) +1 and TAUBnCDRm (slave 2).

An interrupt on slave 2 causes TAUBnTTOUTm of the slave channels to toggle. Depending on the settings of TAUBnTDL.TDLm, delay time is added to positive or negative logic side of the signal (i.e. whether TAUBnTTOUTm toggles immediately or after dead time has elapsed). The duration of the dead time is specified by slave channel 3.

Prerequisites

- Three channels
- The operation mode of the master channel must be set to interval timer mode, see *Table 16-116 “TAUBnCMORm settings for the master channel of the triangle PWM output function with dead time” on page 1130*.

<R>

- Slave channel 1 is not used for this function. This ensures that slave channel 2 is an even channel, and slave channel 3 is an odd channel.
- The operation mode of slave channel 2 must be set to up/down count mode, see *Table 16-120 “TAUBnCMORm settings for slave channel 2 of the triangle PWM output function with dead time” on page 1132*.
  Furthermore, slave channel 2 must be an even channel.
- The operation mode of slave channel 3 must be set to one-count mode, see *Table 16-124 “TAUBnCMORm settings for slave channel 3 of the triangle PWM output function with dead time” on page 1134*.
  Furthermore, slave channel 3 must be an odd channel.
- The channel output mode of the master channel must be set to independent channel output mode 1 (see *16.8 “Channel Output Modes” on page 962*).
- The channel output mode of slave channels 2 and 3 must be set to synchronous channel output mode 2 with dead time output (see *16.8 “Channel Output Modes” on page 962*).
- The following settings establish TAUBnTTOUTm at high level for the down status of the carrier cycle.
  - If the TAUBnCMORm.MD0 (master) bit is set to 0,TAUBnTO.TOm must be set to 1 while TAUBnTOE.TOEm is 0.
  - If the TAUBnCMORm.MD0 (master) bit is set to 1, TAUBnTO.TOm must be set to 0 while TAUBnTOE.TOEm is 0.

Note Slave channel 1 is not used for triangle PWM output function with dead time.

Description The counters are started by setting the channel trigger bits (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm, enabling count operation. The current values of TAUBnCDRm is written to TAUBnCNTm and the counters start to count down from these values. Depending on the setting of the master channel

TAUBnCMORm.MD0 bit an interrupt is generated and TAUBnTTOUTm signal of the master toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles. The counter reloads the TAUBnCDRm value and counts down.

- Slave channel 2:

  The INTTAUBnIm of the master channel triggers the counter of the slave channel 2:

  – If the slave counter currently counts down, it changes count direction.

  – If the slave counter currently counts up, the value of TAUBnCDRm is reloaded and the counter counts down.

  The counter continues to count down or up and awaits the next INTTAUBnIm of the master channel.

- Slave channel 3:

  INTTAUBnIm of slave channel 2 triggers the counter of slave channel 3. The current value of TAUBnCDRm (slave 3) is written to TAUBnCNTm (slave 3) and the counter starts to count down from this value.
  When the counter reaches 0000_H, dead time has elapsed, INTTAUBnIm is generated. The counter returns to $FFFF_H$ and awaits the next INTTAUBnIm of slave channel 2.

An interrupt on slave channel 2 causes TAUBnTTOUTm to toggle as follows:

- It is set by the interrupt when slave channel 2 is counting down
- It is reset by the interrupt when slave channel 2 is counting up.

However, the TAUBnTDL.TDLm settings of the corresponding channel specify whether it is set/reset immediately, or after dead time has elapsed, as shown in *Table 16-115 “Behavior of TAUBnTTOUTm when an interrupt occurs on slave channel 2” on page 1127*.

The TAUBnTOL.TOLm settings specify whether set corresponds to a high signal (TAUBnTOL.TOLm = 0) or a low signal (TAUBnTOL.TOLm = 1).

In triangle PWM output function with dead time, TAUBnTTOUTm can be switched between positive and negative phase by setting TAUBnTOL.TOLm during operation.

The counter can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channel(s) stop but retain their values.

TAUBnCDRm value of slave channel 2 can be set to 0000_H to output 100 % TAUBnTTOUTm.

Note If a forced restart is executed during operation, TAUBnTTOUTm is not output as a triangle PWM signal.

Conditions Simultaneous rewrite can be used with this function. See *16.7 "Simultaneous Rewrite" on page 952*.

TAUBnTOL.TOLm bits should be set before the counter starts, and slave channels 2 and 3 should have opposite TAUBnTOL.TOLm settings or opposite TAUBnTDL.TDLm settings.

Table 16-115 Behavior of TAUBnTTOUTm when an interrupt occurs on slave channel 2

TAUBnTDL.TDLm	Count direction of slave channel 2 when interrupt is generated	TAUBnTTOUTm toggles
0	Down	Set after dead time has elapsed
	Up	Reset immediately
1	Down	Set immediately
	Up	Reset after dead time has elapsed

Notes

1. When the set and reset condition for TAUBnTTOUTm occur simultaneously:
   - If TAUBnTOL.TOLm = 0, the reset condition has priority
   - If TAUBnTOL.TOLm = 1, the set condition has priority.
2. To generate a two-phase PWM output with added dead time, the settings of TAUBnTOL.TOLm bit of slave channels 2 and 3 must not be changed.

(2) Equations

Pulse cycle = (TAUBnCDRm (master) + 1) x count clock cycle

Carrier cycle (down/up) = (TAUBnCDRm (master) + 1) × 2 × count clock cycle

Duty cycle [%] =
[(TAUBnCDRm (master) + 1 - TAUBnCDRm (slave))/ (TAUBnCDRm (master) + 1)] x 100

– Duty cycle = 100 %

  TAUBnCDRm (slave) = 0000_H

– Duty cycle = 0 %

  TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

PWM signal width (positive phase) = [((TAUBnCDRm (master) + 1 - TAUBnCDRm (slave 2) × 2) - (TAUBnCDRm (slave 3) + 1)] × count clock cycle

PWM signal width (negative phase) = [((TAUBnCDRm (master) + 1 - TAUBnCDRm (slave 2) × 2) + (TAUBnCDRm (slave 3) + 1)] × count clock cycle

(3) Block diagram and general timing diagram

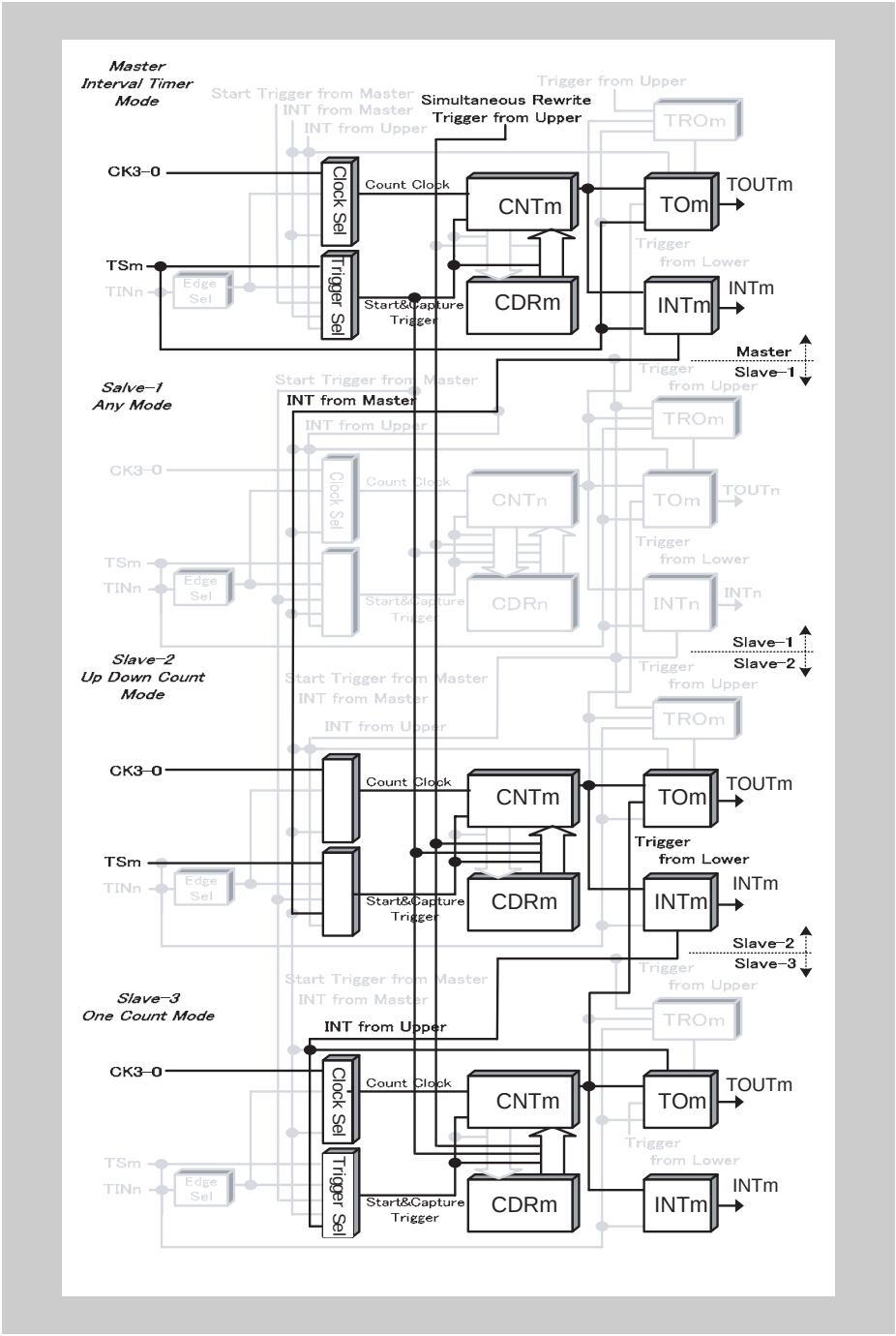


Figure 16-96 Block diagram for triangle PWM output function with dead time

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)
- Slave channel 2:
  - INTTAUBnIm not generated at operation start (TAUBnCMORm.MD0 = 0)
  - TAUBnTDL.TDLm = 0
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)
  - TAUBnTDL.TDLm = 1
  - Negative logic (TAUBnTOL.TOLm = 1)

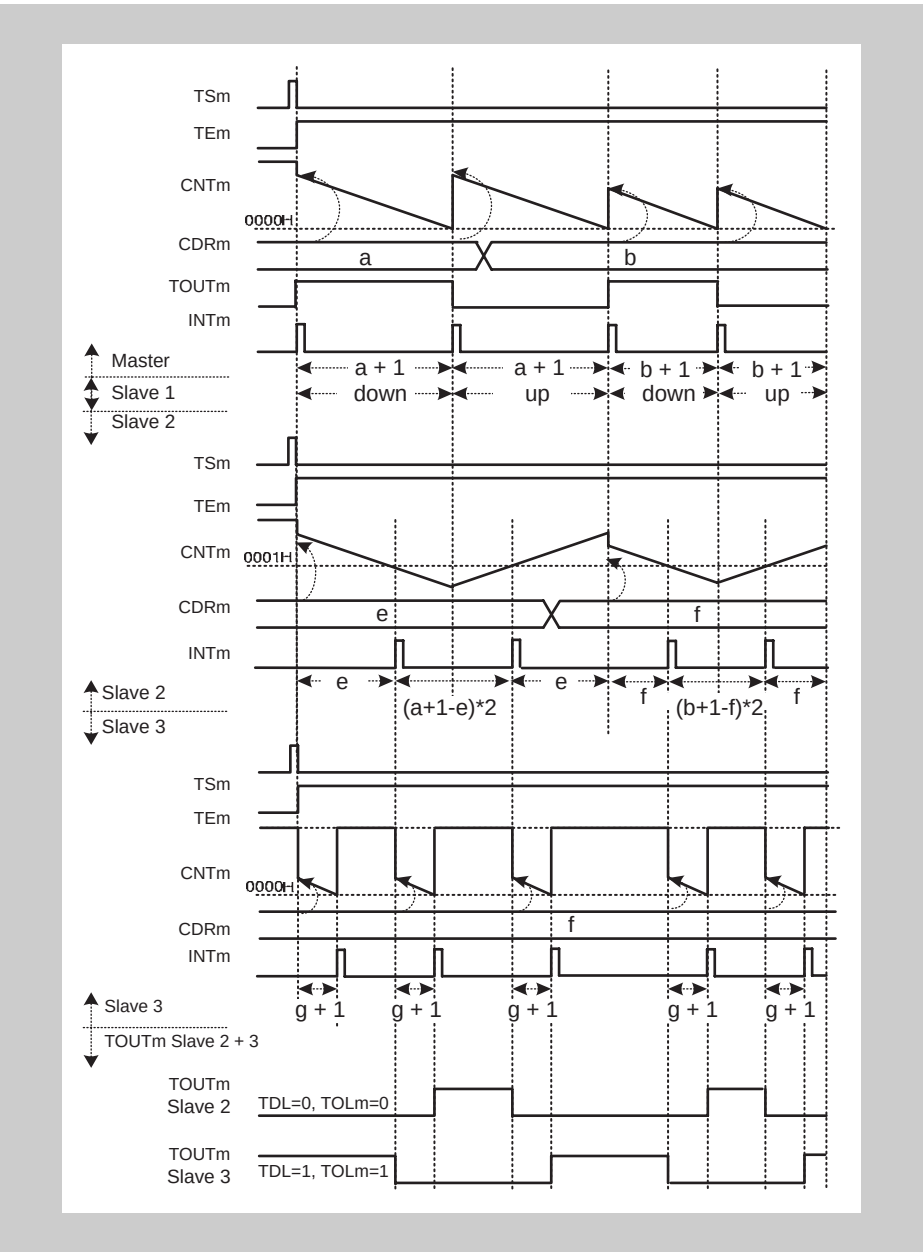


Figure 16-97 General timing diagram for triangle PWM output function with dead time

(4) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-116 TAUBnCMORm settings for the master channel of the triangle PWM output function with dead time

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval timer mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-117 TAUBnCMURm settings for the master channel of the triangle PWM output function with dead time

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

Table 16-118 Control bit settings for independent channel output mode 1

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	0: Independent channel output
TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOLm	0: Positive logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-119 Simultaneous rewrite settings for the master channel of the triangle PWM output function with dead time

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 2

(a) TAUBnCMORm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-120 TAUBnCMORm settings for slave channel 2 of the triangle PWM output function with dead time

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUBnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1001: Up/down count mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-121 TAUBnCMURm settings for slave channel 2 of the triangle PWM output function with dead time

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 2

Table 16-122 Control bit settings for synchronous channel output mode 2 with dead time output

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	1: Synchronous channel operation
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	1: Enables dead time operation
TDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase

(d) Simultaneous rewrite for slave channel 2

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-123 Simultaneous rewrite settings for slave channel 2 of the triangle PWM output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channel 3

(a) TAUBnCMORm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-124 TAUBnCMORm settings for slave channel 3 of the triangle PWM output function with dead time

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUBnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One-count mode
MD0	1: Enables start trigger detection during counting

(b) TAUBnCMURm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-125 TAUBnCMURm settings for slave channel 3 of the triangle PWM output function with dead time

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 3

Table 16-126 Control bit settings for synchronous channel output mode 2 with dead time output

Bit name	Setting
TOEm	1: Disables independent channel output mode controlled by software
TOMm	1: Synchronous channel operation
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	1: Enables dead time operation
TDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase

(d) Simultaneous rewrite for slave channel 3

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-127 Simultaneous rewrite settings for slave channel 3 of the triangle PWM output function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for triangle PWM output function with dead time

Table 16-128 Operating procedure for triangle PWM output function with dead time (1/2)

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 1130*. Slave channel 2: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(5) "Register settings for slave channel 2" on page 1132*. Slave channel 3: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *(6) "Register settings for slave channel 3" on page 1134*. Set the values of the TAUBnCDRm registers of all channels.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)

Table 16-128 Operating procedure for triangle PWM output function with dead time (2/2)

	Operation	Status of TAUBn
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUBnIm is generated on the master channel., depending on the setting of TAUBnCMORm.MD0.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	TAUBnCNTm of the master channel and slave channel 2 load TAUBnCDRm and count down. When the counter of the master channel reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation • TAUBnCNTm (slave 2) reloads the TAUBnCDRm value or counts in the reverse direction When TAUBnCNTm (slave 2) reaches 0001_H: • INTTAUBnIm (slave 2) is generated • TAUBnTTOUTm is reset (TAUBnTOL.TOLm=0) • TAUBnCNTm of slave channel 3 reloads the TAUBnCDRm value and counts down When TAUBnCNTm of slave channel 3 = 0000_H: • INTTAUBnIm is generated • TAUBnTTOUTm is set (TAUBnTOL.TOLm=0) Whether TAUBnTTOUTm (slave 2 and 3) are set/reset with INTTAUBnIm (slave 2) simultaneously or after dead time has elapsed depends on the settings in TAUBnTDL.TDLm and TAUBnTOL.TOLm (see *Table 16-115 "Behavior of TAUBnTTOUTm when an interrupt occurs on slave channel 2" on page 1127*).
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm of slave channels 2 and 3 is initialized to the value set by TAUBnTO.TOm.

Restart

(8) Specific timing diagrams

(a) Duty cycle = 0 %

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

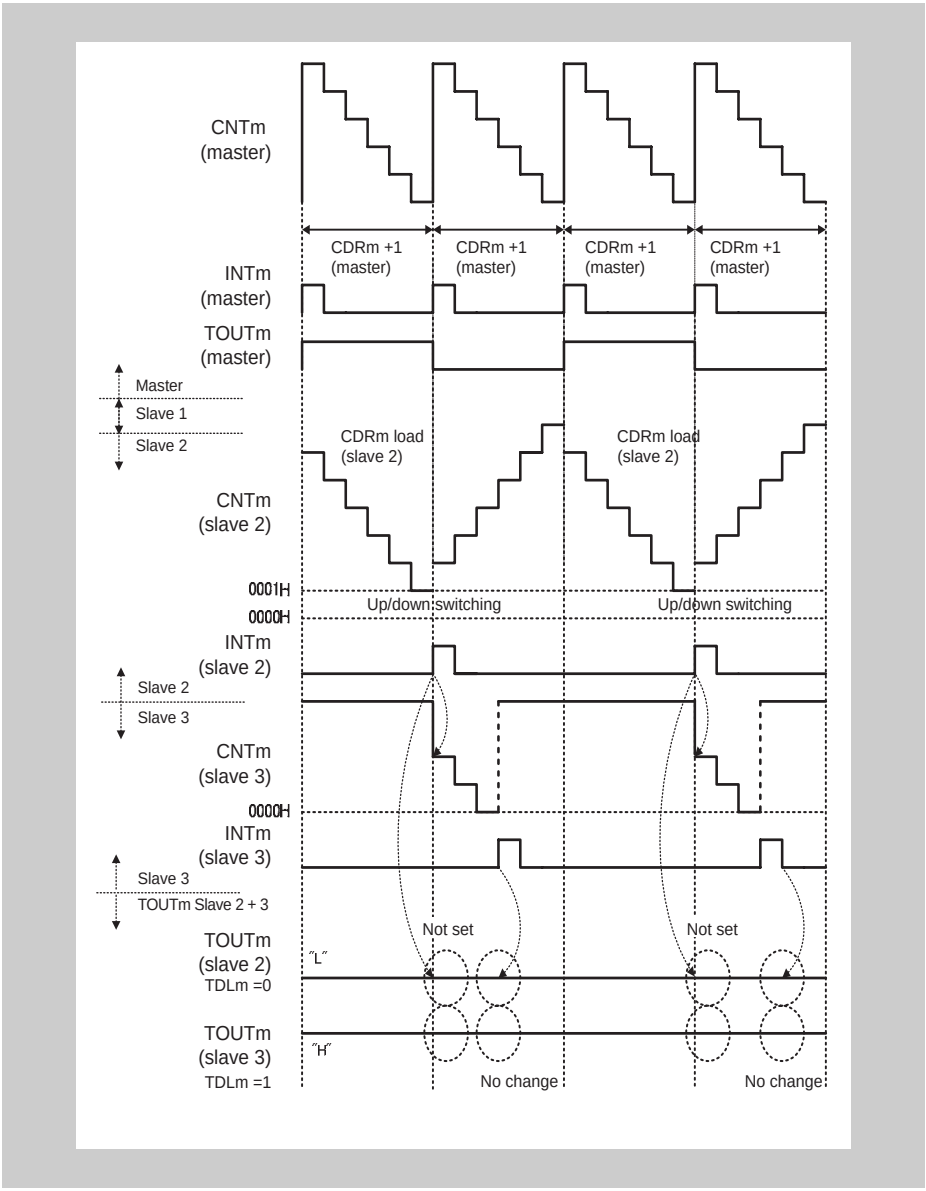


Figure 16-98 TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

- If TAUBnCDRm (slave 2) ≥ TAUBnCDRm (master) the counter of slave channel cannot reach 0000_H during counting down. Therefore TAUBnTTOUTm cannot toggle, i.e. it remains at its initial state. The interrupt from slave channel 2 occurs during count up, therefore it is a reset signal.

(b) Duty cycle = 100 %

The following settings apply to the diagram below:

- Slave channels 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channels 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

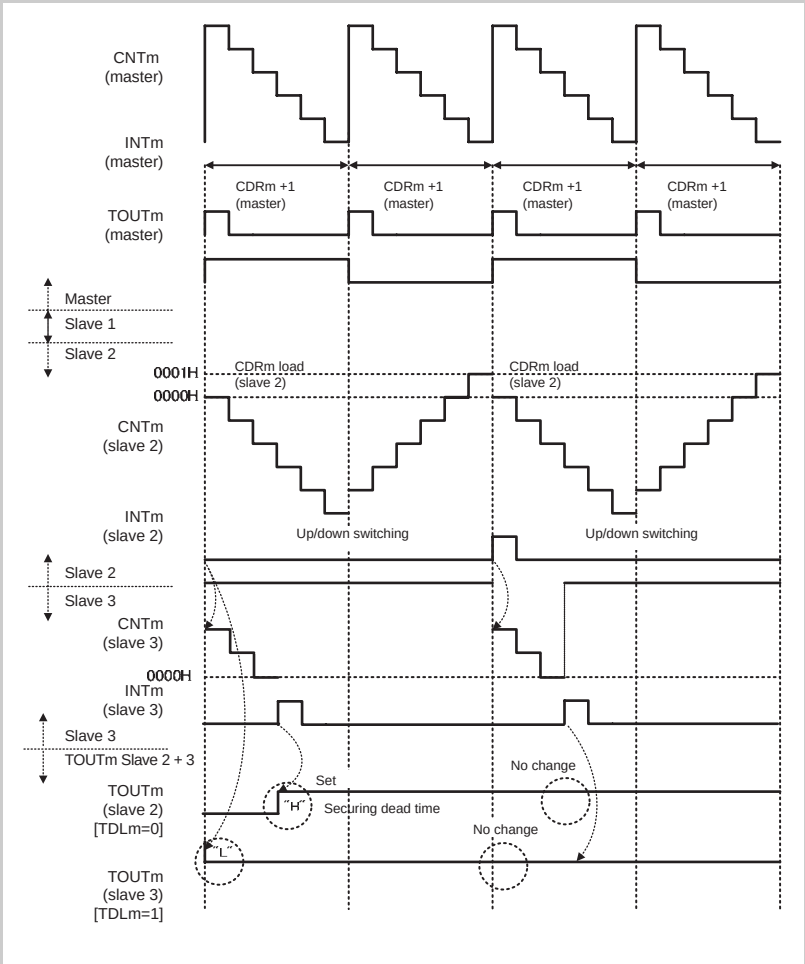


Figure 16-99 TAUBnCDRm (slave) = 0000_H

- If TAUBnCDRm (slave 2) =0000_H the counter of slave channel cannot reach 0001_H while counting up and therefore cannot generate an INTTAUBnIm while counting up.
  - The set conditions for a channel in which TAUBnTDL.TDLm = 0 are met after dead time has elapsed. TAUBnTTOUTm toggles but remains in the new state because the reset conditions never occur for such a channel.
  - Slave channel 3 in the diagram above is set when the counter starts. However, the reset conditions for a channel in which TAUBnTDL.TDLm = 1 never occur so TAUBnTTOUTm remains in its initial state for such a slave channel. In slave 3 in the diagram above, the initial state is high because TAUBnTOL.TOLm = 1.

(c) TAUBnTTOUTm (slave 2) = 0 % and TAUBnTTOUTm (slave 3) > 0 %

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

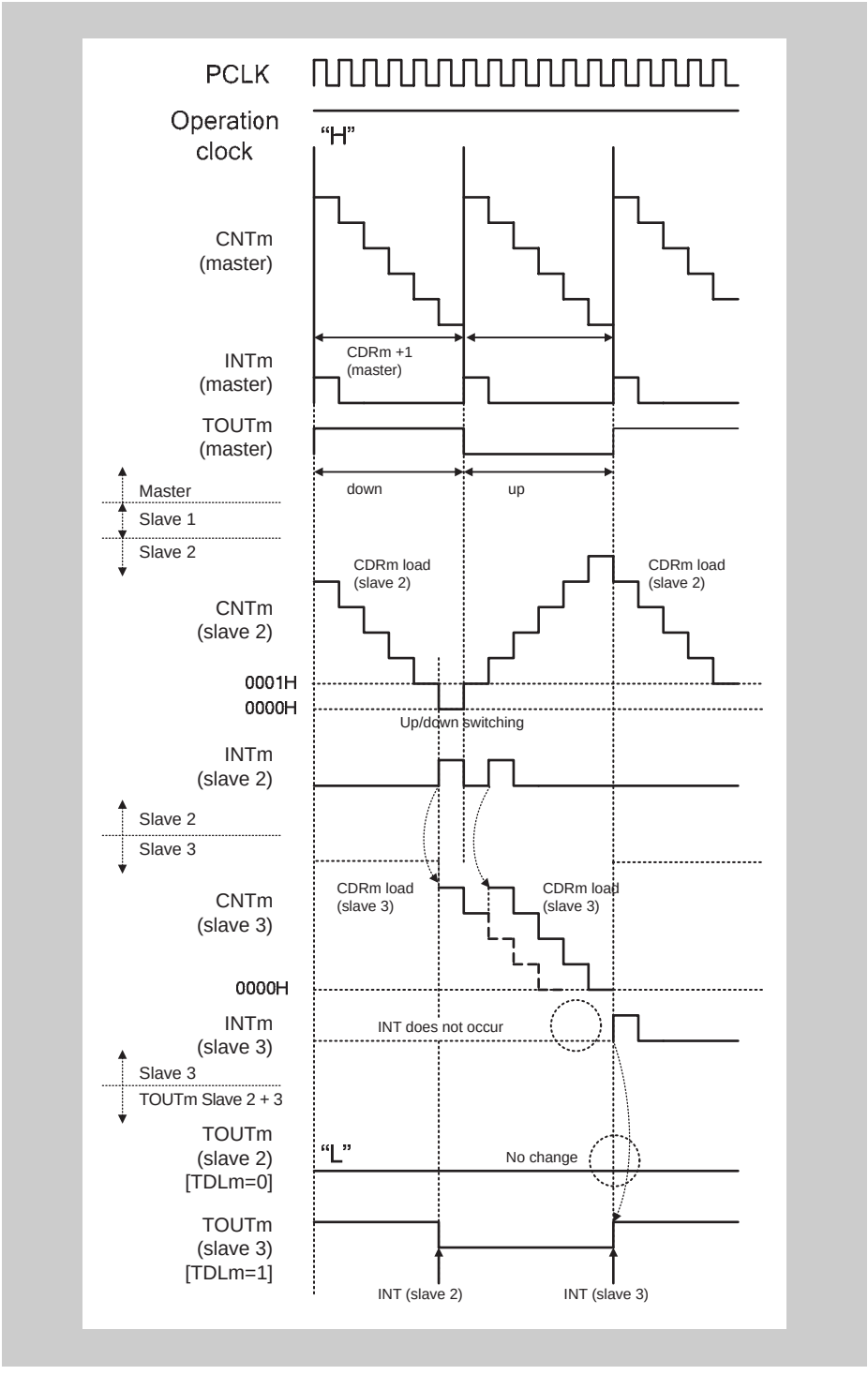


Figure 16-100 TAUBnCDRm (master) = 0005_H, TAUBnCDRm (slave 2) = 0005_H, TAUBnCDRm (slave 3) = 0004_H

- When the counter of slave channel 2 reaches 0000_H, INTTAUBnIm (slave 2) is generated. The counter of slave channel 3 starts to count down.
- If another INTTAUBnIm (slave 2) is generated while the counter of slave channel 3 is still counting down, the value of TAUBnCDRm (slave 3) is reloaded and the counter restarts counting down from this value.
- In the diagram above, the first interrupt on channel 2 occurs while the counter is counting down, and the second whilst it is counting up.
- After the first interrupt, a slave for which TAUBnTDL.TDLm = 0 waits for dead time to elapse before setting. However, before the dead time has elapsed, another interrupt occurs on slave 2, this time while the counter is counting up. This acts as a reset signal, meaning that a channel for which TAUBnTDL.TDLm = 0 always remains inactive.
- TAUBnTTOUTm of a slave channel for which TAUBnTDL.TDLm = 1 is set and reset as normal when the corresponding INTTAUBnIm is generated.

(d) TAUBnTTOUTm (slave 2) > 0 % and TAUBnTTOUTm (slave 3) = 100 %

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

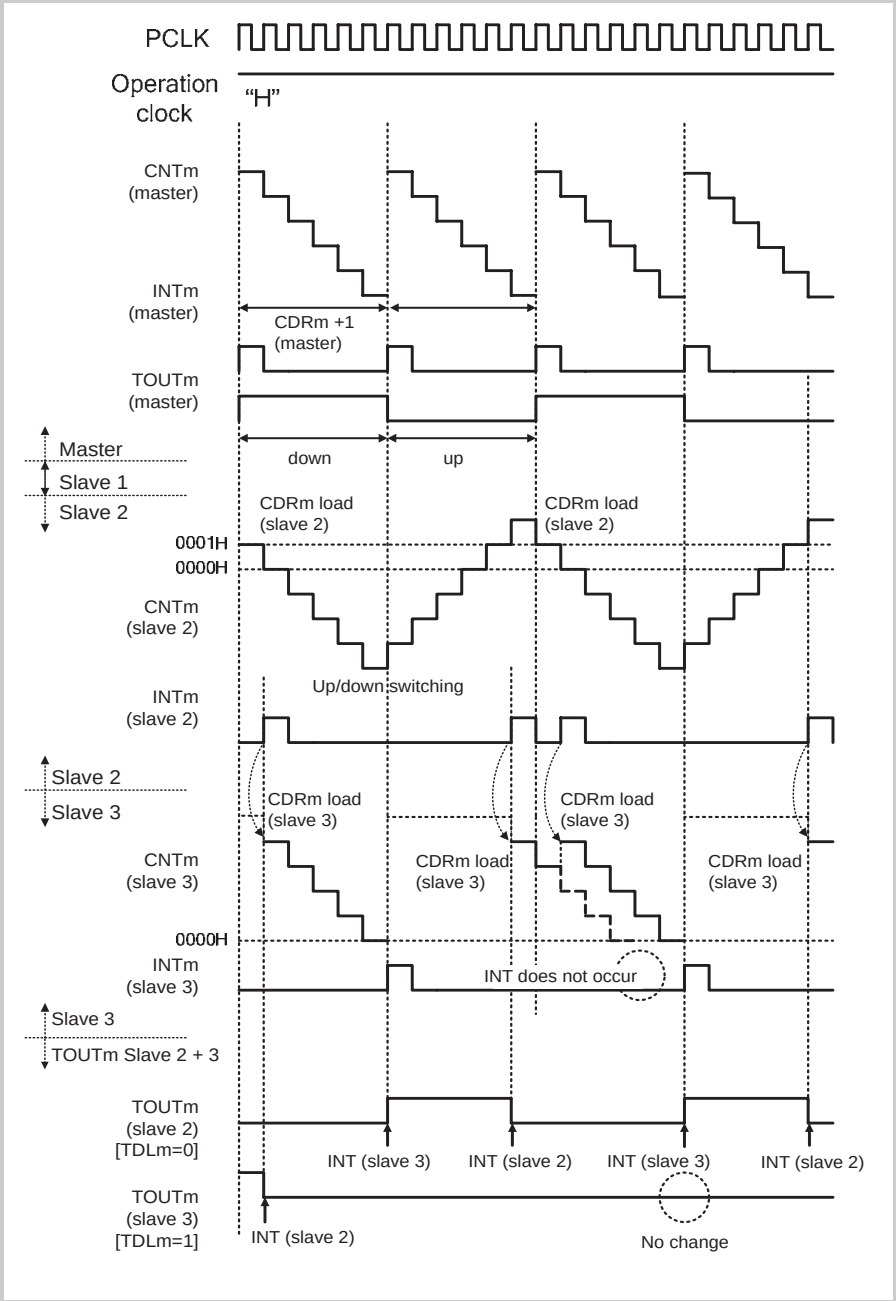


Figure 16-101 TAUBnCDRm (master) = 0005_H, TAUBnCDRm (slave 2) = 0002_H, TAUBnCDRm (slave 3) = 0004_H
PWM signal width (negative phase) ≥ Carrier cycle

- After the second interrupt, a slave for which TAUBnTDL.TDLm = 1 waits for dead time to elapse before resetting. However, before the dead time has elapsed, another interrupt occurs on slave 2, this time while the counter is counting up. This acts as a set signal, meaning that a channel for which TAUBnTDL.TDLm = 1 always remains active.
- TAUBnTTOUTm of a slave channel for which TAUBnTDL.TDLm = 0 is set and reset as normal when the corresponding INTTAUBnIm is generated.

(e) Inhibited INTTAUBnIm to set TAUBnTTOUTm positive phase period

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

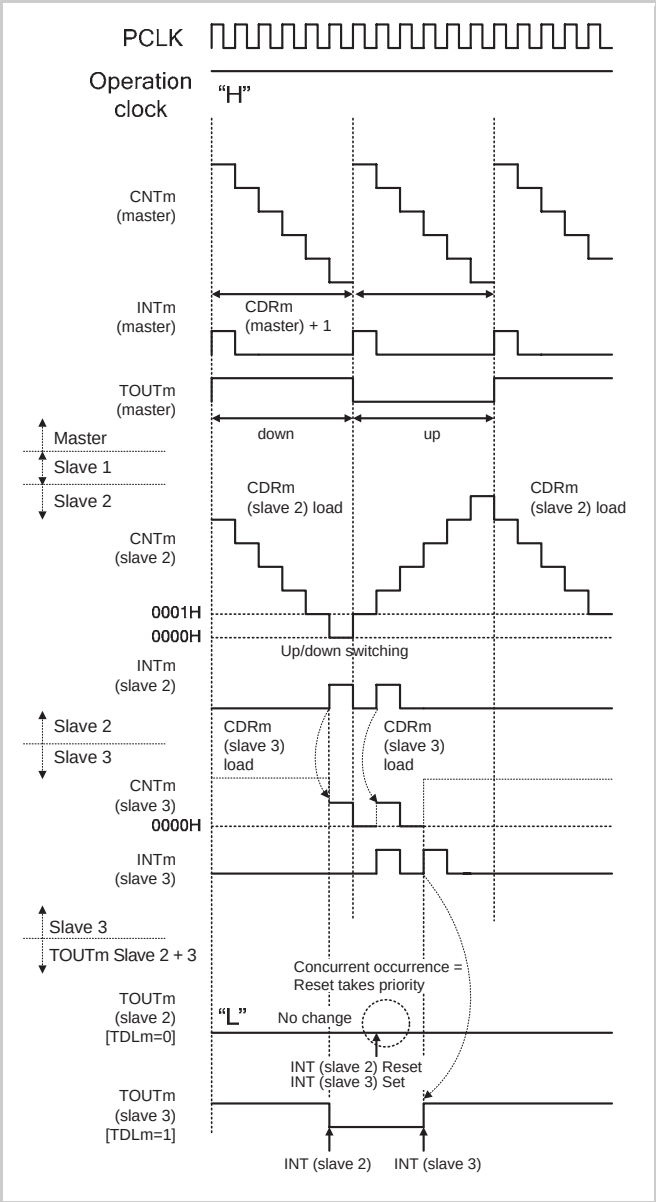


Figure 16-102 TAUBnCDRm (master) = 0005_H, TAUBnCDRm (slave 2) = 0005_H, TAUBnCDRm (slave 3) = 0001_H PWM signal width (positive phase) = 0

- The counter of slave channel 3 reaches 0000_H and generates an INTTAUBnIm to set the TAUBnTTOUTm of slave channel for which TAUBnTDL.TDLm = 0 (slave channel 2 in this example).
- If channel 2 generates an INTTAUBnIm to reset TAUBnTTOUTm simultaneously, this reset signal has priority (assuming TAUBnTOL.TOLm = 0, otherwise the set signal has priority).
- Therefore, TAUBnTTOUTm of a slave channel for which TAUBnTDL.TDLm = 0 remains in its initial state.

(f) Inhibited INTTAUBnIm to set TAUBnTTOUTm negative phase period

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

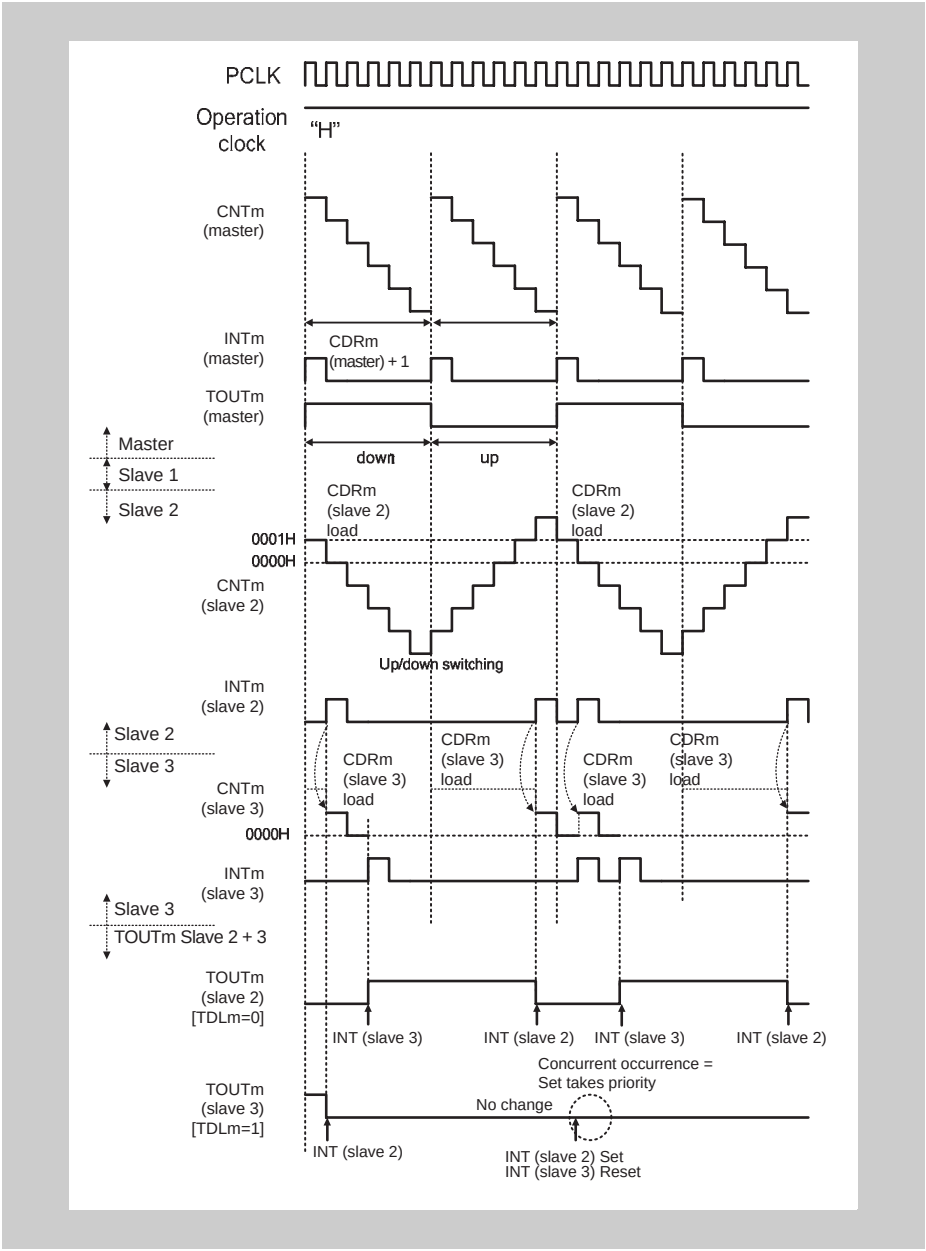


Figure 16-103 TAUBnCDRm (master) = 0005_H, TAUBnCDRm (slave 2) = 0001_H, TAUBnCDRm (slave 3) = 0001_H PWM signal width (negative phase) = carrier cycle

- The counter of slave channel 3 reaches 0000_H and generates an INTTAUBnIm to set the TAUBnTTOUTm of slave channel for which TAUBnTDL.TDLm = 1 (slave 3 in this example).
- If channel 2 generates an INTTAUBnIm to reset TAUBnTTOUTm simultaneously, the set signal has priority (assuming TAUBnTOL.TOLm = 1, otherwise the reset signal has priority).
- Therefore, TAUBnTTOUTm of slave channel for which TAUBnTDL.TDLm = 1 remains in its initial state.

16.21.3 AD conversion trigger output function type 2

(1) Overview

<R> **Summary** This is achieved by setting the channel output mode of the slave to independent channel output mode controlled by software.

(2) Block diagram and general timing diagram

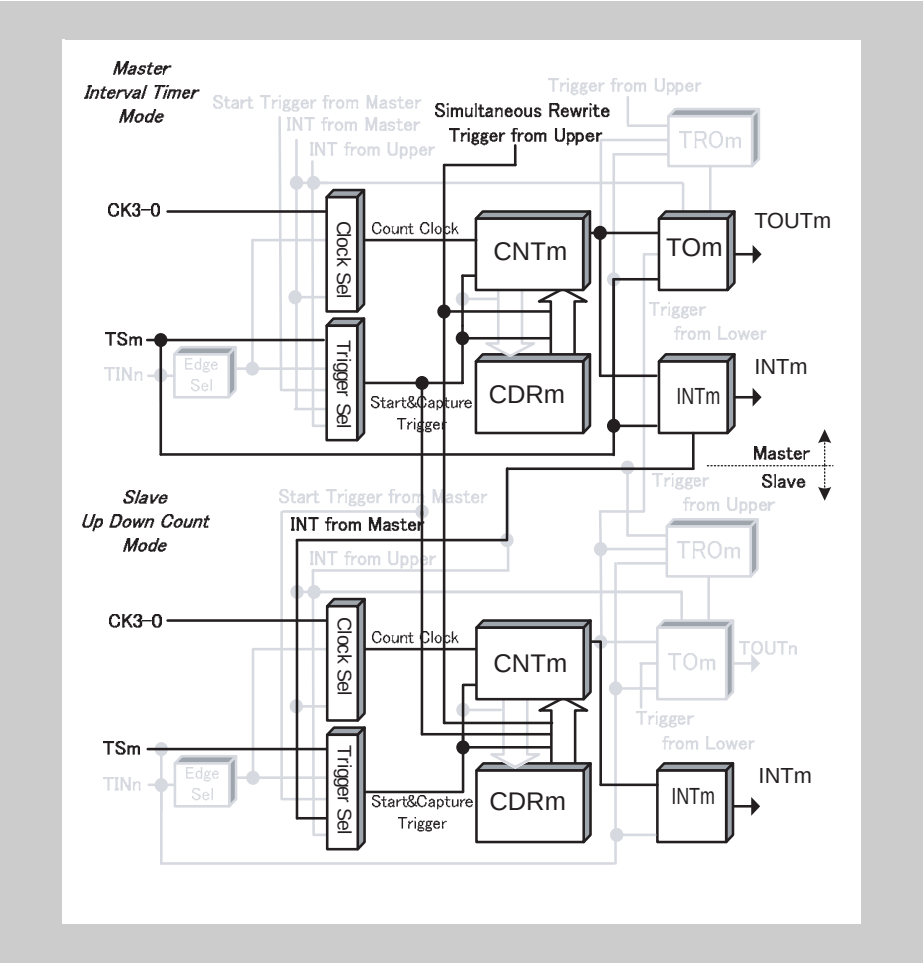


Figure 16-104 Block diagram for AD conversion trigger output function type 2

The following settings apply to the general timing diagram:

- Master channel
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)

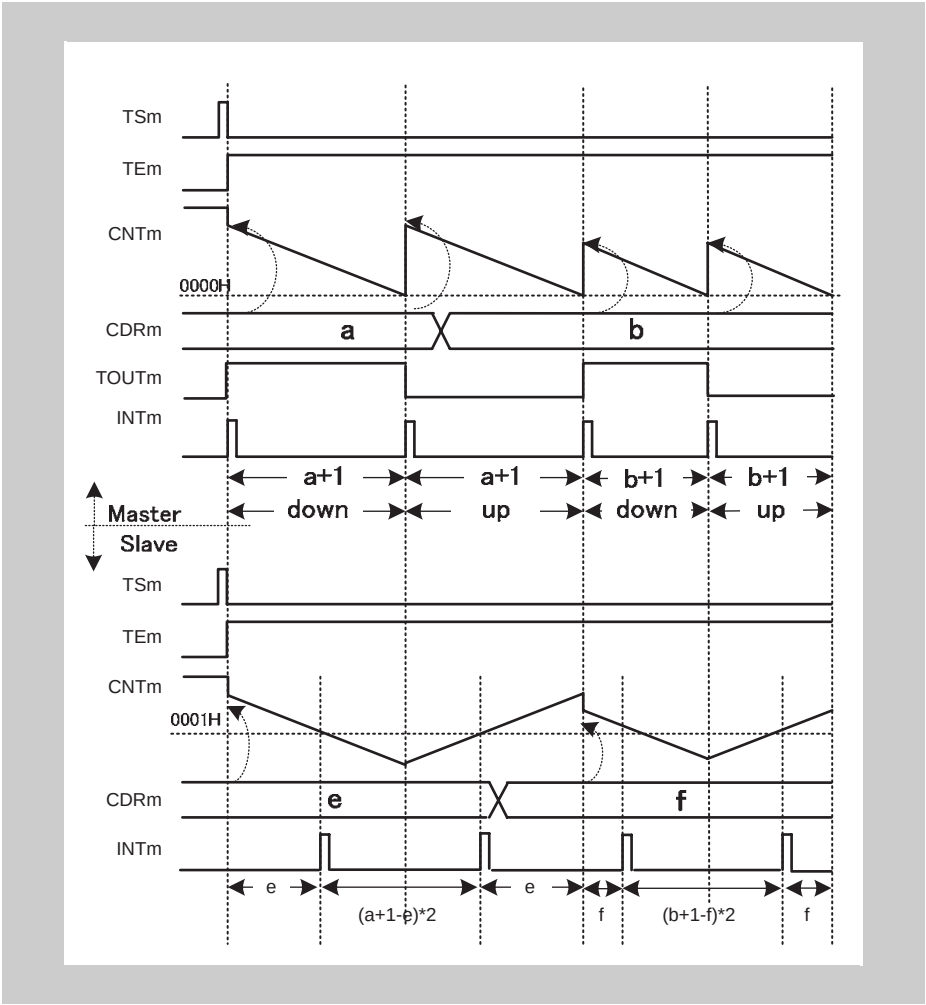


Figure 16-105 General timing diagram for AD conversion trigger output function type 2

16.22 Registers

This section contains a description of all the registers of the 16-bit timer array unit B.

16.22.1 TAUBn register overview

The TAUBn is controlled and operated by the registers in the following table. Where there is one register per channel, this is indicated by an “m”, where m runs from 0 to 15.

Table 16-129 TAUBn register overview

Register name	Symbol	Address
TAUBn prescaler registers		
TAUBn prescaler clock select register	TAUBnTPS	<TAUBn_base> + 240_H
TAUBn control registers		
TAUBn channel data register m	TAUBnCDRm	<TAUBn_base> + m x 4_H
TAUBn channel counter register m	TAUBnCNTm	<TAUBn_base> + 80_H + m x 4_H
TAUBn channel mode OS register m	TAUBnCMORm	<TAUBn_base> + 200_H + m x 4_H
TAUBn channel mode user register m	TAUBnCMURm	<TAUBn_base> + $C0_H$ + m x 4_H
TAUBn channel status register m	TAUBnCSRm	<TAUBn_base> + 140_H + m x 4_H
TAUBn channel status clear trigger register m	TAUBnCSCm	<TAUBn_base> + 180_H + m x 4_H
TAUBn channel start trigger register	TAUBnTS	<TAUBn_base> + $1C4_H$
TAUBn channel enable status register	TAUBnTE	<TAUBn_base> + $1C0_H$
TAUBn channel stop trigger register	TAUBnTT	<TAUBn_base> + $1C8_H$
TAUBn output registers		
TAUBn channel output enable register	TAUBnTOE	<TAUBn_base> + $5C_H$
TAUBn channel output register	TAUBnTO	<TAUBn_base> + 58_H
TAUBn channel output mode register	TAUBnTOM	<TAUBn_base> + 248_H
TAUBn channel output configuration register	TAUBnTOC	<TAUBn_base> + $24C_H$
TAUBn channel output active level register	TAUBnTOL	<TAUBn_base> + 040_H
TAUBn channel dead time output enable register	TAUBnTDE	<TAUBn_base> + 250_H
TAUBn channel dead time output level register	TAUBnTDL	<TAUBn_base> + 54_H
TAUBn reload data registers		
TAUBn channel reload data enable register	TAUBnRDE	<TAUBn_base> + $260C_H$
TAUBn channel reload data mode register	TAUBnRDM	<TAUBn_base> + 264_H
TAUBn channel reload data control CH select register	TAUBnRDS	<TAUBn_base> + 268_H
TAUBn channel reload data control register	TAUBnRDC	<TAUBn_base> + $26C_H$
TAUBn channel reload data trigger register	TAUBnRDT	<TAUBn_base> + 44_H
TAUBn channel reload status register	TAUBnRSF	<TAUBn_base> + 48_H

<TAUBn_base> The <TAUBn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

16.22.2 TAUBn prescaler register details

(1) TAUBnTPS - TAUBn prescaler clock select register

This register specifies the PCLK prescalers for clocks CK0, CK1, CK2, and CK3 for all channels.

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + 240_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRS3[3:0]				PRS2[3:0]				PRS1[3:0]				PRS0[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-130 TAUBnTPS register contents (1/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15 to 12</td><td>PRS3[3:0]</td><td>Specifies the CK3 clock.
<table>
<tr><th>PRS3[3:0]</th><th>CK3 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK3 are stopped (TAUBnTE.TEm = 0).</td></tr>
<tr><td>11 to 8</td><td>PRS2[3:0]</td><td>Specifies the CK2 clock.
<table>
<tr><th>PRS2[3:0]</th><th>CK2 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK2 are stopped (TAUBnTE.TEm = 0).</td></tr>
</table>

Table 16-130 TAUBnTPS register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7 to 4</td><td>PRS1[3:0]</td><td>Specifies the CK1 clock.
<table>
<tr><th>PRS1[3:0]</th><th>CK1 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK1 are stopped (TAUBnTE.TEm = 0).</td></tr>
<tr><td>3 to 0</td><td>PRS0[3:0]</td><td>Specifies the CK0 clock.
<table>
<tr><th>PRS0[3:0]</th><th>CK0 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK0 are stopped (TAUBnTE.TEm = 0).</td></tr>
</table>

Note The TAUBn clock input PCLK is specified in the first section of this chapter under the keyword “Clock supply”.

16.22.3 TAUBn control register details

(1) TAUBnCDRm - TAUBn channel data register

This register functions either as a compare register or as a capture register, depending on the operation mode specified in TAUBnCMORm.MD[4:0].

Access This register can be read/written in 16-bit units.

- In capture mode, only reading is possible. Write operation is ignored.
- In compare mode, reading and writing is possible.

Address <TAUBn_base> + 0_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDR[15:0]															
R/W															

Table 16-131 TAUBnCDRm register contents

Bit position	Bit name	Function
15 to 0	CDR[15:0]	Data register for the capture/compare value.

(2) TAUBnCNTm - TAUBn channel counter register

This register is the channel m counter register.

Access This register can be read in 16-bit units.

Address <TAUBn_base> + 80_H + m x 4_H

Initial Value 0000_H or $FFFF_H$ The initial value depends on the operation mode, see *Table 16-133 "TAUBnCNTm read values after the counter is re-enabled" on page 1154*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R															

Table 16-132 TAUBnCNTm register contents

Bit position	Bit name	Function
15 to 0	CNT[15:0]	16-bit counter value.

The read value depends on the counter, the operation mode change, and the values of the TAUBnTS.TSm and TAUBnTT.TTm bits.

The *initial* counter read value depends on the operation mode and how the counter was stopped:

- by a reset
- by a counter stop trigger (TAUBnTT.TTm = 1)

The following table lists the initial counter read values after the counter has stopped (TAUBnTE.TEm= 0) and re-enabled (TAUBnTS.TSm = 1).

The table also contains the counter read value one count after the counter is enabled (TAUBnTS.TSm = 1) for modes where the counter waits for a start trigger.

Table 16-133 TAUBnCNTm read values after the counter is re-enabled

Mode name	Count method (up/down)	TAUBnCNTm value		
		After reset	After stop trigger	After one count
Interval timer mode	Count down	$FFFF_H$	Stop value	-
Judge mode	Count down	$FFFF_H$	Stop value	-
Capture mode	Count up	0000_H	Stop value	-
Event count mode	Count down	$FFFF_H$	Stop value	-
One-count mode	Count down	$FFFF_H$	Stop value	$FFFF_H$
Capture & one-count mode	Count up	0000_H	Stop value	Captured value + 1 (TAUBnCDRm)
Judge & one-count mode	Count down	$FFFF_H$	Stop value	TAUBnCNTm value - 1
Up/down count mode	Count up/down	$FFFF_H$	Stop value	-
Pulse one-count mode	Count down	$FFFF_H$	Stop value	0000_H
Count capture mode	Count up	0000_H	Stop value	-
Gate count mode	Count down	$FFFF_H$	Stop value	Stop value
Capture & gate count mode	Count up	0000_H	Stop value	Stop value

Note If the operation mode is changed while the counter is stopped, the initial counter value after counter restart is undefined. The operation mode is changed by register TAUBnCMORm.MD[4:0].

(3) TAUBnCMORm - TAUBn channel mode OS register

This register controls channel m operation. It specifies, for example, the operation clock, count clock, and master/slave function.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUBnTE.TEm = 0).

Address <TAUBn_base> + 200_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:0]				
R/W		R	R/W	R/W	R/W			R/W		R	R/W				

Table 16-134 TAUBnCMORm register contents (1/3)

Bit position	Bit name	Function
15, 14	CKS[1:0]	Selects the operation clock. The operation clock is used for the TAUBnTTINm input edge detection circuit. It can also be used as the count clock depending on bit TAUBnCMORm.CCS0.
12	CCS0	Selects the count clock for TAUBnCNTm counter:
11	MAS	Specifies the channel as master or slave channel during synchronous channel operation: 0: Slave 1: Master This bit is only valid for even channels (CHm_even). For odd channels (CHm_odd), it is fixed to 0.

CKS1	CKS0	Selected operation clock
0	0	CK0
0	1	CK1
1	0	CK2
1	1	CK3

<R>

CCS0	Selected count clock
0	Operation clock as specified by TAUBnCMORm.CKS[1:0]
1	Valid edge of TAUBnTTINm input signal

Table 16-134 TAUBnCMORm register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>10 to 8</td><td>STS[2:0]</td><td>Selects the external start trigger:
<table>
<tr><th>STS2</th><th>STS1</th><th>STS0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Software trigger</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Valid edge of the TAUBnTTINm input signal. TAUBnCMURm.TIS[1:0] specifies the valid edge.</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Valid edge of the TAUBnTTINm input signal is the start trigger and the reverse edge is the stop (capture) trigger</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>INTTAUBnI of the master channel</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>INTTAUBnI of the upper channel (m-1), regardless of the master setting</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>Dead-time output signal TAUBnTTDL of the TAUBnTTOUTm generation unit</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>Up/down output trigger signal TAUBnTUDSm of the master channel.</td></tr>
</table></td></tr>
<tr><td>7, 6</td><td>COS[1:0]</td><td>Specifies when the capture register TAUBnCDRm and the overflow flag TAUBnCSRm.OVF of channel m are updated.
These bits are only valid if channel m is in capture mode.
<table>
<tr><th>COS1</th><th>COS0</th><th>Capture register</th><th>TAUBnCSRm.OVF</th></tr>
<tr><td>0</td><td>0</td><td rowspan="2">Updated upon detection of a TAUBnTTINm input valid edge.</td><td>Updated (cleared or set) upon detection of a TAUBnTTINm input valid edge:
• If a counter overflow has occurred since the last valid edge detection, TAUBnCSRm.OVF is set.
• If no counter overflow has occurred since the last valid edge detection, TAUBnCSR.OVF is cleared.</td></tr>
<tr><td>0</td><td>1</td><td>Set upon counter overflow and cleared by a CPU instruction.</td></tr>
<tr><td>1</td><td>0</td><td rowspan="2">Updated upon detection of a TAUBnTTINm input valid edge and upon counter overflow:
• TAUBnTTINm input valid edge: Counter value is written to TAUBnCDRm
• Overflow: $FFFF_H$ is written to TAUBnCDRm. The next TAUBnTTINm input valid edge detection is ignored.</td><td>Not set.</td></tr>
<tr><td>1</td><td>1</td><td>Set upon counter overflow and cleared by a CPU instruction.</td></tr>
</table></td></tr>
</table>

Table 16-134 TAUBnCMORm register contents (3/3)

Bit position	Bit name	Function
4 to 0	MD[4:0]	Specifies the operation mode.

MD4	MD3	MD2	MD1	MD0	Description
0	0	0	0	1/0	Interval timer mode
0	0	0	1	1/0	Judge mode
0	0	1	0	1/0	Capture mode
0	0	1	1	0	Event count mode
0	1	0	0	1/0	One-count mode
0	1	0	1	1/0	Setting prohibited
0	1	1	0	0	Capture & one-count mode
0	1	1	1	1/0	Judge & one-count mode
1	0	0	0	0	Setting prohibited
1	0	0	1	0	Up/down count mode
1	0	1	0	1/0	Pulse one-count mode
1	0	1	1	1/0	Count capture mode
1	1	0	0	0	Gate count mode
1	1	0	1	0	Capture & gate count mode

Mode	Role of the MD0 bit
Interval timer mode Capture mode Count capture mode	Specifies whether an INTTAUBnIm is generated when the counter is triggered: 0: No INTTAUBnIm generated 1: INTTAUBnIm generated
Event count mode Up/down count mode	This bit must be set to 0: 0: No INTTAUBnIm generated when the counter is triggered 1: -
One-count mode Gate count mode	Enables/disables start trigger detection during counting: 0: Disabled 1: Enabled INTTAUBnIm and TAUBnTTOUTm are not output when the counter is triggered.
Pulse one-count mode	Enables/disables start trigger detection during counting: 0: Disabled 1: Enabled INTTAUBnIm and TAUBnTTOUTm are output when the counter is triggered.
Capture & one-count mode Capture & gate count mode	This bit must be set to 0: 0: No INTTAUBnIm generated when the counter is triggered. Start trigger is disabled during counting. 1: -
Judge mode Judge one-count mode	Specifies when INTTAUBnIm is generated: 0: When TAUBnCNTm ≤ TAUBnCDRm 1: When TAUBnCNTm > TAUBnCDRm

(4) TAUBnCMURm - TAUBn channel mode user register

This register specifies the type of valid edge detection used for the TAUBnTTINm input.

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + $C0_H$ + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	TIS[1:0]	
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W

Table 16-135 TAUBnCMURm register contents

Bit position	Bit name	Function
1, 0	TIS[1:0]	Specifies the valid edge of the TAUBnTTINm input: TIS1 / TIS0 / Description 0 / 0 / Falling edge 0 / 1 / Rising edge 1 / 0 / Rising and falling edges (low-width measurement selection). Start trigger: falling edge. Stop trigger (capture): rising edge 1 / 1 / Rising and falling edges (high-width measurement selection). Start trigger: rising edge. Stop trigger (capture): falling edge. This setting is only valid if the start trigger selection is set to TAUBnCMORm.STS[2:0] = 010_B • To detect rising and falling edges when TAUBnCMORm.STS[2:0] is not set to 010_B, set TAUBnCMURm.TIS[1:0] = 10_B. • Edge detection for TAUBnTTINm input signals is performed based on the operation clock selected by TAUBnCMORm.CKS[1:0].

(5) TAUBnCSRm - TAUBn channel status register

This register indicates the count direction and the overflow status of channel m's counter.

Access This register can be read in 16-bit units.

Address <TAUBn_base> + 140_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	CSF	OVF
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 16-136 TAUBnCSRm register contents

Bit position	Bit name	Function
1	CSF	Indicates the count direction: 0: Counts up 1: Counts down The read value of this bit is only valid in the following mode: • Up/down count mode For channel 0 this bit is fixed to 0.
0	OVF	Indicates the counter overflow status: 0: No overflow occurred 1: Overflow occurred This bit is only used in the following modes: • Capture mode • Capture & one-count mode • Count capture mode • Capture & gate count mode The function of this bit depends on the setting of control bits TAUBnCMORm.COS[1:0]. OVF is not be set when TAUBnCMORm.COS[1:0] = 10_B.

(6) TAUBnCSCm - TAUBn channel status clear register

This register is a trigger register for clearing the overflow flag TAUBnCSRm.OVF of a channel m.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUBn_base> + 180_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	CLOV
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Table 16-137 TAUBnCSCm register contents

Bit position	Bit name	Function
0	CLOV	0: No function 1: Clears the overflow flag TAUBnCSRm.OVF

(7) TAUBnTS - TAUBn channel start trigger register

This register enables the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUBn_base> + $1C4_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS 15	TS 14	TS 13	TS 12	TS 11	TS 10	TS 09	TS 08	TS 07	TS 06	TS 05	TS 04	TS 03	TS 02	TS 01	TS 00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 16-138 TAUBnTS register contents

Bit position	Bit name	Function
15 to 0	TSm	Enables the counter for channel m: 0: No function 1: Enables the counter and sets TAUBnTE.TEm = 1. When the counter is enabled, this bit immediately returns to 0. TAUBnTE.TEm = 1 only *enables* counter. Whether the counter *starts* depends on the selected operation mode.

(8) TAUBnTE - TAUBn channel enable status register

This register indicates whether counter is enabled or disabled.

Access This register can be read in 16-bit units.

Address <TAUBn_base> + $1C0_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE15	TE14	TE13	TE12	TE11	TE10	TE09	TE08	TE07	TE06	TE05	TE04	TE03	TE02	TE01	TE00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 16-139 TAUBnTE register contents

Bit position	Bit name	Function
15 to 0	TEm	Indicates whether counter for channel m is enabled or disabled: 0: Counter disabled 1: Counter enabled Setting TAUBnTS.TSm to 1 or trigger input detection TAUBnTSSTm = 1 sets this bit to 1. Setting TAUBnTT.TTm to 1 resets this bit to 0.

(9) TAUBnTT - TAUBn channel stop trigger register

This register stops the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUBn_base> + $1C8_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TT15	TT14	TT13	TT12	TT11	TT10	TT09	TT08	TT07	TT06	TT05	TT04	TT03	TT02	TT01	TT00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 16-140 TAUBnTT register contents

Bit position	Bit name	Function
15 to 0	TTm	Stops the counter of channel m: 0: No function 1: Stops the counter and sets TAUBnTE.TEm = 0. When the counter has stopped, this bit immediately returns to 0. TAUBnCNTm stops counting and TAUBnCNTm, TAUBnTO.TOm, and TAUBnTTOUTm all retain the values they had before the counter was stopped.

16.22.4 TAUBn output register details

(1) TAUBnTOE - TAUBn channel output enable register

This register enables and disables independent channel output mode controlled by software.

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + $5C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOE 15	TOE 14	TOE 13	TOE 12	TOE 11	TOE 10	TOE 09	TOE 08	TOE 07	TOE 06	TOE 05	TOE 04	TOE 03	TOE 02	TOE 01	TOE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-141 TAUBnTOE register contents

Bit position	Bit name	Function
15 to 0	TOEm	Enables/disables independent channel output mode: 0: Enables independent channel output mode 1: Disables independent channel output mode

(2) TAUBnTOM - TAUBn channel output mode register

This register specifies the output mode of each channel.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUBnTE.TEm = 0).

Address <TAUBn_base> + 248_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM 15	TOM 14	TOM 13	TOM 12	TOM 11	TOM 10	TOM 09	TOM 08	TOM 07	TOM 06	TOM 05	TOM 04	TOM 03	TOM 02	TOM 01	TOM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-142 TAUBnTOM register contents

Bit position	Bit name	Function
15 to 0	TOMm	Specifies the channel output mode: 0: Independent channel output mode 1: Synchronous channel output mode The output mode depends on several channel output control bits, as can be seen in *Table 16-10 “Channel output modes” on page 963*.

(3) TAUBnTOC - TAUBn channel output configuration register

This register specifies the output mode of each channel in combination with TAUBnTOMm.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUBnTE.TEm = 0).

Address <TAUBn_base> + $24C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOC 15	TOC 14	TOC 13	TOC 12	TOC 11	TOC 10	TOC 09	TOC 08	TOC 07	TOC 06	TOC 05	TOC 04	TOC 03	TOC 02	TOC 01	TOC 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-143 TAUBnTOC register contents

Bit position	Bit name	Function
15 to 0	TOCm	Specifies the output mode: 0: Operation mode 1 1: Operation mode 2 The output mode also depends on TAUBnTOM.TOMm, as can be seen in the following table.

TOMm	TOCm	Description
0	0	Toggle mode: TAUBnTTOUTm toggles when INTTAUBnIm occurs.
	1	Set/reset mode: TAUBnTTOUTm set when INTTAUBnIm occurs upon count start and reset when INTTAUBnIm occurs due to detection of a match between TAUBnCNTm and TAUBnCDRm, or, in one-shot pulse output function when the counter is 0.
1	0	Synchronous channel operation mode 1: TAUBnTTOUTm set when INTTAUBnI occurs on the master channel and reset when INTTAUBnI occurs on the slave channel.
	1	Synchronous channel operation mode 2: TAUBnTTOUTm set when INTTAUBnIm occurs while the slave channel is counting down and reset when INTTAUBnIm occurs while the slave channel is counting up.

(4) TAUBnTDE - TAUBn channel dead time output enable register

This register enables/disables dead time operation for each channel.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUBnTE.TEm = 0).

Address <TAUBn_base> + 250_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDE 15	TDE 14	TDE 13	TDE 12	TDE 11	TDE 10	TDE 09	TDE 08	TDE 07	TDE 06	TDE 05	TDE 04	TDE 03	TDE 02	TDE 01	TDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-144 TAUBnTDE register contents

Bit position	Bit name	Function
15 to 0	TDEm	Enables/disables dead time control operation of channel m: 0: Disables dead time operation 1: Enables dead time operation The same settings must be set for the even and the odd slave channel that comprise a set. These bits only apply when: • TAUBnTOE.TOEm, TAUBnTOM.TOMm, and TAUBnTOC.TOCm = 1.

(5) TAUBnTDL - TAUBn channel dead time output level register

This register selects the phase period to which dead time is added.

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + 54_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDL 15	TDL 14	TDL 13	TDL 12	TDL 11	TDL 10	TDL 09	TDL 08	TDL 07	TDL 06	TDL 05	TDL 04	TDL 03	TDL 02	TDL 01	TDL 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-145 TAUBnTDL register contents

Bit position	Bit name	Function
15 to 0	TDLm	Selects the phase period to which dead time is added: 0: Positive phase period 1: Negative phase period These bits only apply when: • TAUBnTOE.TOEm, TAUBnTOM.TOMm, TAUBnTOC.TOCm, and TAUBnTDE.TDEm = 1.

16.22.5 TAUBn channel output level register details

(1) TAUBnTO - TAUBn channel output register

This register specifies and reads the level of TAUBnTTOUTm.

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + 58_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO15	TO14	TO13	TO12	TO11	TO10	TO09	TO08	TO07	TO06	TO05	TO04	TO03	TO02	TO01	TO00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-146 TAUBnTO register contents

Bit position	Bit name	Function
15 to 0	TOm	Specifies/reads the level of TAUBnTTOUTm: 0: Low 1: High Only TOm bits for which Independent Channel Output function is disabled (TAUBnTOEm = 0) can be written.

(2) TAUBnTOL - TAUBn channel output level register

This register specifies the output logic of the channel output bit (TAUBnTO.TOm).

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + 040_H

Initial Value 00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOL 15	TOL 14	TOL 13	TOL 12	TOL 11	TOL 10	TOL 09	TOL 08	TOL 07	TOL 06	TOL 05	TOL 04	TOL 03	TOL 02	TOL 01	TOL 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-147 TAUBnTOL register contents

Bit position	Bit name	Function
15 to 0	TOLm	Specifies the output logic of the channel m output bit (TAUBnTO.TOm): 0: Positive logic (active high) 1: Inverted logic (active low) These bits apply in all channel output modes except independent channel output mode controlled by software and channel output modes with real-time output.

16.22.6 TAUBn simultaneous rewrite register details

(1) TAUBnRDE - TAUBn channel reload data enable register

This register enables and disables simultaneous rewrite of the data register TAUBnCDRm. It also enables simultaneous rewrite of the data register TAUBnTOLm for the PWM output function and the triangle PWM output function.

Access This register can be read/written in 16-bit units. It can only be written when TAUBnTE.TEm = 0.

Address <TAUBn_base> + 260_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDE 15	RDE 14	RDE 13	RDE 12	RDE 11	RDE 10	RDE 09	RDE 08	RDE 07	RDE 06	RDE 05	RDE 04	RDE 03	RDE 02	RDE 01	RDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-148 TAUBnRDE register contents

Bit position	Bit name	Function
15 to 0	RDEm	Enables/disables simultaneous rewrite of the data register of channel m: 0: Disables simultaneous rewrite 1: Enabled simultaneous rewrite

(2) TAUBnRDM - TAUBn channel reload data mode register

This register selects when the signal that controls simultaneous rewrite is loaded.

Access This register can be read/written in 16-bit units. It can only be written when TAUBnTE.TEm = 0.

Address <TAUBn_base> + 264_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDM 15	RDM 14	RDM 13	RDM 12	RDM 11	RDM 10	RDM 09	RDM 08	RDM 07	RDM 06	RDM 05	RDM 04	RDM 03	RDM 02	RDM 01	RDM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-149 TAUBnRDM register contents

Bit position	Bit name	Function
15 to 0	RDMm	Selects when the signal that triggers simultaneous is generated: 0: When the master channel counter starts counting 1: At the top of a triangle wave cycle These bits only apply when TAUBnRDE.RDEm = 1 and TAUBnRDS.RDSm = 0.

(3) TAUBnRDS - TAUBn channel reload data control channel select register

This register selects the control channel for simultaneous rewrite.

Access This register can be read/written in 16-bit or 1-bit units. It can only be written when TAUBnTE.TEm = 0.

Address <TAUBn_base> + 268_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDS 15	RDS 14	RDS 13	RDS 12	RDS 11	RDS 10	RDE 09	RDE 08	RDE 07	RDE 06	RDE 05	RDE 04	RDE 03	RDE 02	RDE 01	RDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-150 TAUBnRDS register contents

Bit position	Bit name	Function
15 to 0	RDSm	Specifies which channel is monitored for the simultaneous rewrite trigger: 0: Master channel 1: Another upper channel

(4) TAUBnRDC - TAUBn channel reload data control register

This register specifies the channel that generates the INTTAUBnIm signal that triggers simultaneous rewrite.

Access This register can be read/written in 16-bit units. It can only be written when TAUBnTE.TEm = 0

Address <TAUBn_base> + $26C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDC 15	RDC 14	RDC 13	RDC 12	RDC 11	RDC 10	RDC 09	RDC 08	RDC 07	RDC 06	RDC 05	RDC 04	RDC 03	RDC 02	RDC 01	RDC 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-151 TAUBnRDC register contents

Bit position	Bit name	Function
15 to 0	RDCm	<R> Specifies whether the channel generates the simultaneous rewrite trigger signal. 0: Do not have the channel generate the simultaneous rewrite trigger signal. 1: Have the channel generate the simultaneous rewrite trigger signal. <R> These bits only apply when TAUBnRDS.RDSm = 1.

(5) TAUBnRDT - TAUBn channel reload data trigger register

This register triggers the simultaneous rewrite pending state.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUBn_base> + 044_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDT 15	RDT 14	RDT 13	RDT 12	RDT 11	RDT 10	RDT 09	RDT 08	RDT 07	RDT 06	RDT 05	RDT 04	RDT 03	RDT 02	RDT 01	RDT 00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 16-152 TAUBnRDT register contents

Bit position	Bit name	Function
15 to 0	RDTm	Triggers the simultaneous rewrite pending state: 0: No function 1: Simultaneous rewrite pending state is triggered. The simultaneous rewrite pending flag (TAUBnRSFm) is set to 1. The system waits for the simultaneous rewrite trigger. TAUBnRDT.RDTm immediately returns to 0.

(6) TAUBnRSF - TAUBn channel reload status register

This flag register indicates that simultaneous rewrite is possible.

Access This register can be read in 16-bit units.

Address <TAUBn_base> + 048_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSF 15	RSF 14	RSF 13	RSF 12	RSF 11	RSF 10	RSF 09	RSF 08	RSF 07	RSF 06	RSF 05	RSF 04	RSF 03	RSF 02	RSF 01	RSF 00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 16-153 TAUBnRSF register contents

Bit position	Bit name	Function
15 to 0	RSFm	Indicates the simultaneous rewrite status: 0: Simultaneous rewrite disabled 1: Simultaneous rewrite enabled

Chapter 17 Timer Array Unit J (TAUJ)

This chapter describes timer array unit J (TAUJ).

The first section describes all V850E2/Sx4-H specific properties, such as instances, register base addresses, and input/output signal names.
The subsequent sections describe the features that apply to all implementations.

17.1 V850E2/Sx4-H TAUJ Features

Instances

This microcontroller has following number of instances of TAUJ:

Table 17-1 Instances of TAUJ

TAUJ	
Number of nstances	1
Name	TAUJ0

Instances index n

Throughout this chapter, the individual instances of TAUJ is identified by the index “n” (n = 0), for example, TAUJnTOM for the TAUJn channel output mode register.

Channel index m

TAUJ has 4 channels. Throughout this chapter, the individual channels are identified by the index “m” (m = 0 to 3), thus a certain channel is denoted as CHm.
The even numbered channels (m = 0, 2) are denoted as CHm_even.
The odd numbered channels (m = 1, 3) are denoted as CHm_odd.

Register addresses

All TAUJ n register addresses are given as addresses offset from the individual base address <TAUJn_base>.
The <TAUJn_base> address of each TAUJn is listed in the following table:

Table 17-2 Register base addresses <TAUJn_base>

TAUJn instance	<TAUJn_base> address
TAUJ0	FF81 1000_H

Clock supply

The following clock is supplied to TAUJ:

Table 17-3 TAUJn clock supply

TAUJn	Clock	Connected to:
TAUJ0	PCLK	Clock generator CKSCLK_A03

Interrupts and DMA

TAUJ can generate the following interrupts and DMA requests:

Table 17-4 TAUJn interrupts and DMA requests

TAUJn signals	Function	Connected to:
TAUJ0:		
INTTAUJ0I0	Channel 0 interrupt	Interrupt controller INTTAUJ0I0 DMA controller trigger 59
INTTAUJ0I1	Channel 1 interrupt	Interrupt Controller INTTAUJ0I1 DMA controller trigger 60
INTTAUJ0I2	Channel 2 interrupt	Interrupt Controller INTTAUJ0I2 DMA controller trigger 61
INTTAUJ0I3	Channel 3 interrupt	Interrupt Controller INTTAUJ0I3 DMA controller trigger 62

TAUJ hardware reset

TAUJ and its registers are initialized by the following reset signal:

Table 17-5 TAUJn reset signal

TAUJn	Reset signal
TAUJ0	System reset SYSRES

I/O signals

The I/O signals of TAUJ are listed in the following table.

Table 17-6 TAUJn I/O signals

TAUJ signal	Function	Connected to:
TAUJ0:		
TAUJ0TTIN0 and TAUJ0TTIN1	Channel 0 and 1 inputs	Ports TAUJ0I0 and TAUJ0I1
TAUJ0TTIN2	Channel 2 input	Port TAUJ0I2 or RTCA1HZ[a]
TAUJ0TTIN3	Channel 3 input	Port TAUJ0I3 or RTCA1HZ[a]
TAUJ0TTOUT0 to TAUJ0TTOUT3	Channel 0 to 3 outputs	Ports TAUJ0O0 to TAUJ0O3

a) See *17.2 “TAUJ Input Selection”*.

17.2 TAUJ Input Selection

17.2.1 TAUJ0 input selection

As shown in *Figure 17-1 "Selecting the signal input to TAUJ0"* below, the signal input to TAUJ0 can be captured by using the 1-Hz pulse output function (RTCAT1HZ).

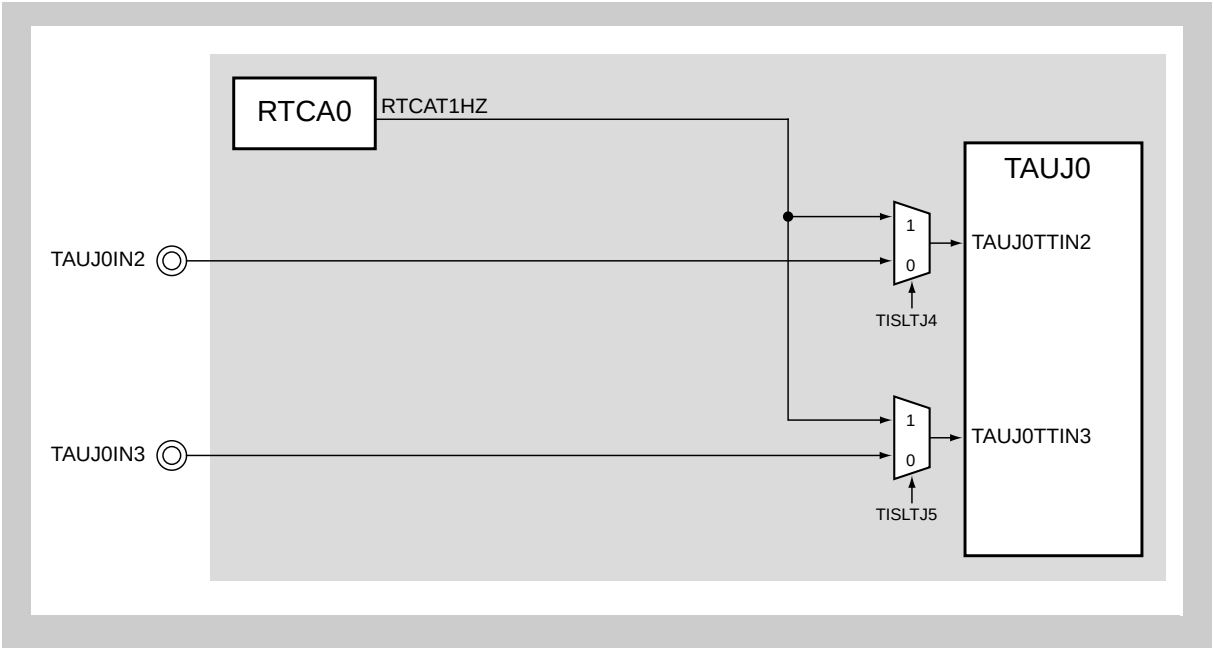


Figure 17-1 Selecting the signal input to TAUJ0

The table below shows how to select the input signal to TAUJ0:

Table 17-7 TAUJ0 input selections

TAUJn input	Input options	Selection control
TAUJ0TTIN2	Port TAUJ0I2	TISLTJ.TISLTJ4 = 0
	RTCA0 RTCAT1HZ (Real-Time Clock 1 Hz output)	TISLTJ.TISLTJ4 = 1
TAUJ0TTIN3	Port TAUJ0I3	TISLTJ.TISLTJ5 = 0
	RTCA0 RTCAT1HZ (Real-Time Clock 1 Hz output)	TISLTJ.TISLTJ5 = 1

(1) TISLTJ - Timer input selection register J

This register selects the TAUJ0 input signal.

Access This register can be read/written in 8-bit units.

Address FF77 3000$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
0	0	TISLTJ5	TISLTJ4	0	0	0	0
R	R	R/W	R/W	R	R	R	R

Table 17-8 TISLTJ register contents

Bit position	Bit name	Function
5	TISLTJ5	Select the signal input to TAUJ0TTIN3. 0: Port TAUJ0I3 1: RTCA0 RTCAT1HZ
4	TISLTJ4	Select the signal input to TAUJ0TTIN2. 0: Port TAUJ0I2 1: RTCA0 RTCAT1HZ

17.3 Functional Overview

Features summary

The TAUJ has the following functions:

- 4 channels
- 32-bit counter and 32-bit data register per channel
- Independent channel operation
- Synchronous channel operation (master and slave operation)
- Generation of different types of output signal
- Counter can be triggered by external signal
- Interrupt generation

The following figure shows the main components of the TAUJ:

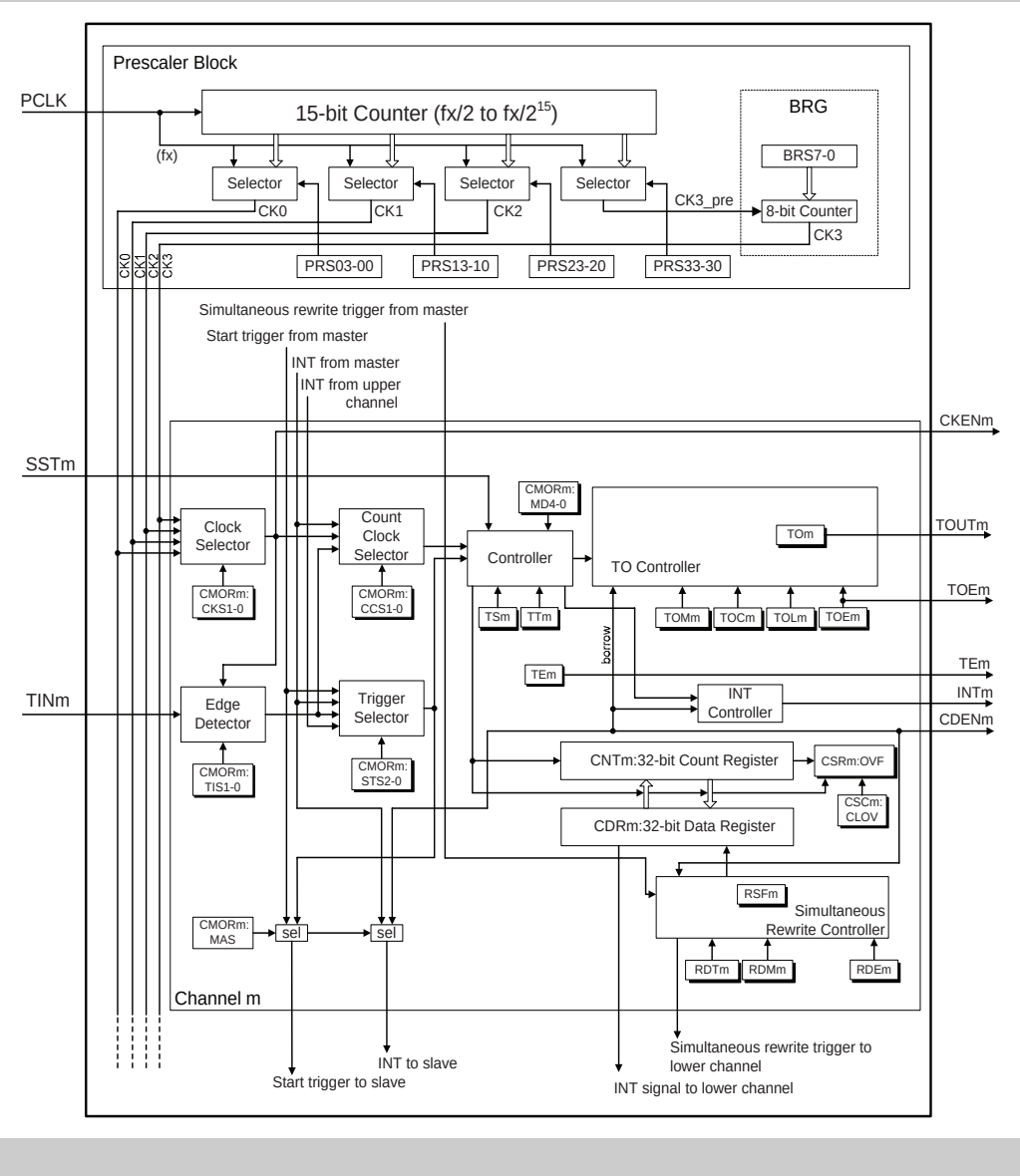


Figure 17-2 Block diagram of the TAUJ

The prefix *TAUJn* has been omitted from the register and block names in the above figure for the sake of clarity.

17.3.1 Terms

In this section, the following terms are used:

- **Independent / synchronous channel operation**

  Independent or synchronous channel operation describes the dependency of channels on each other:

  – If a channel operates independent of all other channels, this is called independent channel operation.

  – If a channel operates depending on other channels, this is called synchronous channel operation.

- **Channel group**

  In synchronous channel operation, all channels that depend on each other are referred to as a “channel group”.
  A channel group has one master channel and one or more slave channels.

- **Operation mode**

  An operation mode can be selected for every channel m. The operation mode defines the *basic* operation and features of a channel.
  In synchronous channel operation, every channel in the channel group can operate in a different operation mode.
  Examples are capture mode and interval timer mode.

- **Channel output mode**

  The channel output mode defines the operation of TAUJnTTOUTm

  – of a single channel (independent output operation) or

  – of all channels in a channel group (synchronous output operation).

  An example is independent channel output mode 1.

- **Channel operation function**

  The channel operation function defines the *complete* function and all features

  – of a single channel (independent channel operation) or

  – of all channels in a channel group (synchronous channel operation).

- **Upper / lower channel**

  Depending on the channel number m, a neighboring channel can be referred to as “upper” or “lower” channel:

  – Upper channel: Channel that has a smaller number

  – Lower channel: Channel that has a greater number

  Example:
  For channel 2, channel 1 is an upper channel and channel 3 is a lower channel.

17.4 Functional Description

TAUJ is used to perform various count or timer operations and to output a signal which depends on the result of the operation. It contains one prescaler for count clock generation and 4 channels, each equipped with a 32-bit counter TAUJnCNTm and a 32-bit data register TAUJnCDRm to hold the start or compare value of the counter.

It also contains several control and status registers.

Independent and synchronous operation

Every channel can operate in two operation modes, either independently or in combination with other channels (synchronously). If a channel group has one master channel and one or more slave channels, the slave channels depend on the master channel.

When a channel is operated independently, its operation mode and functions are not affected by those of other channels. When a channel is operated synchronously it is either a master or a slave. A master channel can have multiple slaves, and the state of one channel affects that of the other channels. For example, a channel can be used to control the timing for starting counting and resetting for another channel.

The following describes the functional blocks:

Prescaler

The prescaler provides up to 4 clock signals (CK0 to CK3) that can be used as count clocks for all channels.

For count clocks CK0 to CK2, a clock obtained by dividing PCLK by 2^0 to 2^{15} using the prescaler can be selected. For the fourth count clock CK3, a division factor that is not a power of 2 can be specified by using BRG.

Clock and count clock selection

For every channel, the count clock selector selects which of the following is used as the clock source:

- One of the clocks CK0 to CK3 (selected by the clock selector)
- INTTAUJnIm from master channel
- TAUJnTTINm input signal valid edge

Controller

The controller controls the main operations of the counter:

- Operation mode (selected by bits TAUJnCMORm.TAUJnMD[4:0])
- Counter start enable (TAUJnTS.TAUJnTSm) and counter stop (TAUJnTT.TAUJnTTm)

  When counter start is enabled, status flag TAUJnTE.TAUJnTEm is set.

Trigger selector

If the counter is enabled (TAUJnTE.TAUJnTEm = 1), it starts automatically or waits for an external start trigger signal, depending on the selected operation mode. Any of the following signals can be used as the start trigger:

- Synchronous channel start trigger input TAUJnTSSTm
  TAUJnTTINm input signal valid edge
- INTTAUJnIm from the master channel

Simultaneous rewrite controller Simultaneous rewrite control is a function that can be used in synchronous operation modes. The data registers of all channels in a channel group (TAUJnCDRm) can be rewritten at any time. The simultaneous rewrite controller ensures that new data register values of all channels become effective at the same time.

TAUJnTO controller The output control of every channel enables the generation of various output signal forms such as PWM signals.

17.4.1 Timer operation functions

The functions below can be achieved by operating TAUJ independently on each channel or by operating it on a combination of multiple channels.

Table 17-9 TAUJ operation functions

Independent channel operation function	Synchronous channel operation function
Independent channel operation functions	Synchronous channel operation function
Interval timer function	PWM output function
TAUJnTTINm input interval timer function	–
Independent channel signal measurement functions	–
Overflow interrupt output function (during TAUJnTTINm width measurement)	–
TAUJnTTINm input period count detection function	–
Overflow interrupt output function (during TAUJnTTINm input period count detection)	–
TAUJnTTINm input pulse interval judgment function	–
TAUJnTTINm input signal width judgment function	–
Other independent channel function	–
TAUJnTTINm input position detection function	–

17.5 General Operating Procedure

The following lists the general operation procedure for the TAUJn:

After reset release, the operation of each channel is stopped. Writing to each register is enabled when clock supply is started. The control register of TAUJnTTOUTm is initialized and outputs a low level.

1. Set the TAUJnTPS and TAUJnBRS registers to specify the clock frequency of CK0 to CK3.
2. Configure the desired TAUJn function:
   – Set the operation mode
   – Set the channel output mode
   – Set any other control bits
3. Enable the counter by setting the TAUJnTS.TAUJnTSm bit to 1.

   The counter starts counting immediately, or when an appropriate trigger is detected, depending on the bit settings.
4. During counting, if desired, and if possible for the configured function, stop the counter or perform a forced restart operation.
5. Stop the function by setting the TAUJnTT.TAUJnTTm bit to 1.

Note A detailed description of the required control bits and the operation of the individual functions is given in the following sections:

17.15 “Independent Channel Interrupt Functions” on page 1202

17.16 “Independent Channel Signal Measurement Functions” on page 1216

17.17 “Other Independent Channel Function” on page 1246

17.6 Operation Modes

The TAUJ contains 7 operation modes.

One operation mode can be set for each channel. It is specified using the TAUJnCMORm.TAUJnMD[4:0] bits.

Note For registers and bits, some values are fixed according to the operation function, and others are selected by the user.

For details about the settings for registers and bits, see the sections describing each operation function.

17.7 Concepts of Synchronous Channel Operation

In synchronous channel operation, multiple channels depend on each other, or are affected by changes in another channel. Therefore, several rules apply for the use of synchronous channel functions. These rules are detailed in *17.7.1 “Rules”*.

Two special features for synchronous channel operation are detailed in the following subsections:

- *17.7.2 “Simultaneous start and stop of synchronous channel counters” on page 1182*
- *17.8 “Simultaneous Rewrite” on page 1183*

17.7.1 Rules

Number of masters and slaves

- Only even channels (CH0, CH2) can be set as master channels. Any channel except CH0 can be set as a slave channel.
- Only channels lower than the master channel can be set as slave channels, and several slave channels can be set for one master channel.
  Example: If CH2 is a master channel, CH3 can be set as slave channel.
- If two master channels are used, slave channels cannot cross the master channels.
  Example: If CH0 and CH2 are master channels, CH1 can be set as a slave channel for CH0, but CH3 cannot.

Operation clock

- The same operation clock must be set for the corresponding slave channel and the master channel. Specify the same value for the TAUJnCMORm.TAUJnCKS[1:0] bits of the synchronized master and slave channels.

The basic concepts of master/slave usage and operation clocks are illustrated in the following figure.

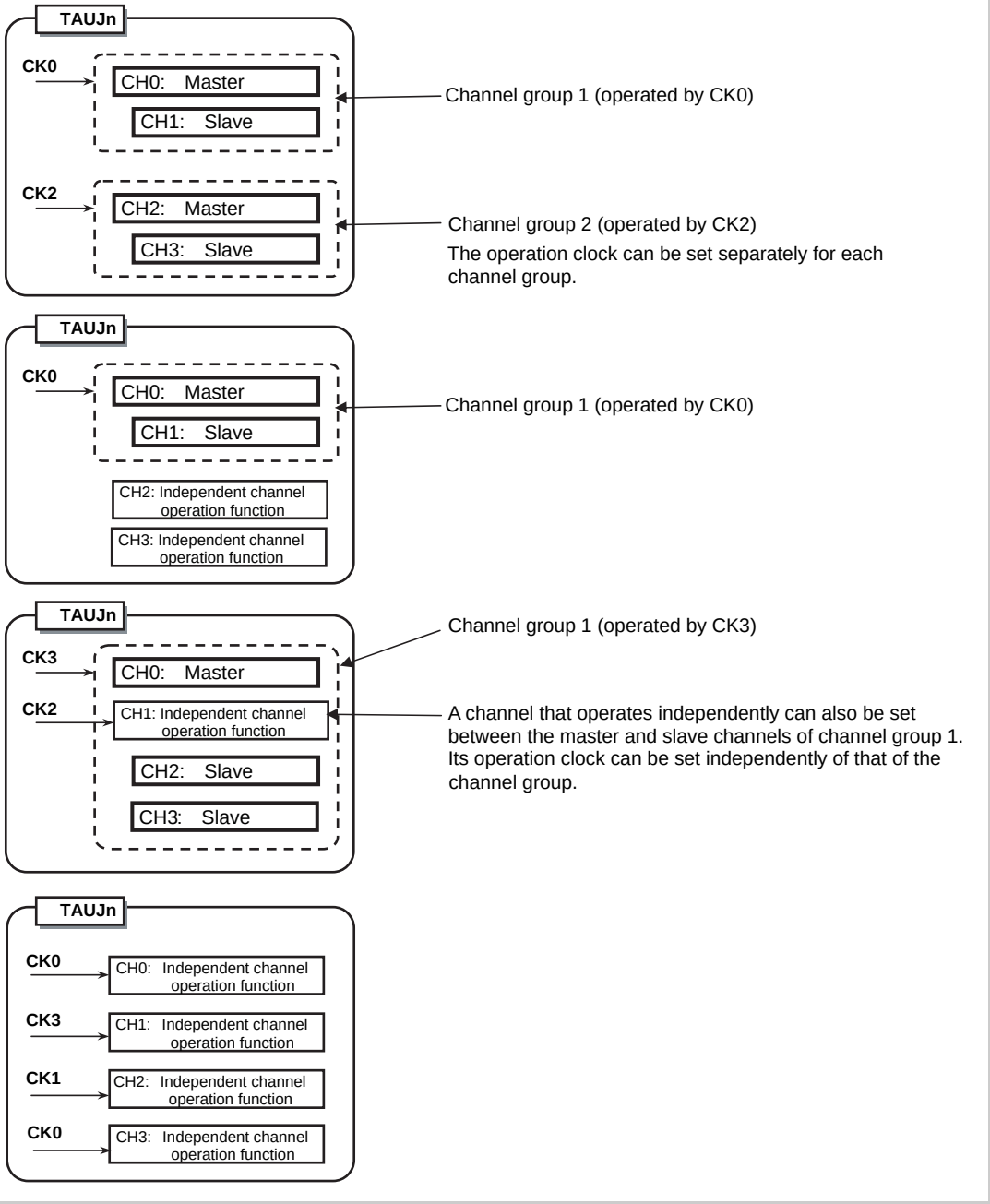


Figure 17-3 Grouping of the channels and assignment of operation clocks

Control trigger signals of the master and slave channels

- A master channel can output a control trigger signal to its slave channels.
- Slave channels can use the control trigger signal of the master channel but cannot transfer their control trigger signals to the lower channels.
- A master channel cannot use the control trigger signal of an upper master channel.

17.7.2 Simultaneous start and stop of synchronous channel counters

Channels that are operated synchronously can be started and stopped simultaneously, both within a unit, and between units.

(1) Simultaneous start and stop within a unit

- To simultaneously start synchronized channels, the TAUJnTS.TAUJnTSm bits of the channels must be set at the same time.
- To simultaneously stop synchronized channels, the TAUJnTT.TAUJnTTm bits of the channels must be set at the same time.

Setting a TAUJnTS.TAUJnTSm bit to 1 sets the corresponding TAUJnTE.TAUJnTEm bit to 1, enabling counting. When the counter starts counting depends on the operation mode.

(2) Simultaneous start between units

Counters in different units can also be started simultaneously if the corresponding counters are enabled before receiving the simultaneous trigger signal.

17.8 Simultaneous Rewrite

17.8.1 Introduction

Simultaneous rewrite describes the ability to change the compare/start value and the output logic of multiple channels at the same time.

The corresponding data and control registers (TAUJnCDRm and TAUJnTOLm) can be written at any time. The new value does not affect the counter operation or the output signal until simultaneous rewrite is triggered.

Simultaneous rewrite is triggered when the counter on the master channel reaches a certain value.

The following table shows the settings for simultaneous rewrite (TAUJnRDM.TAUJnRDMm = 0).

Table 17-10 Simultaneous rewrite settings

Method	Simultaneous rewrite triggered when	TAUJnRDE.TAUJnRDEm
-	No simultaneous rewrite	0
A	The master channel (re)starts counting	1

17.8.2 How to control simultaneous rewrite

The following figure shows the general procedure for simultaneous rewrite. The three main blocks (Initial settings, Start counter & count operation, and Simultaneous rewrite) are explained afterwards.

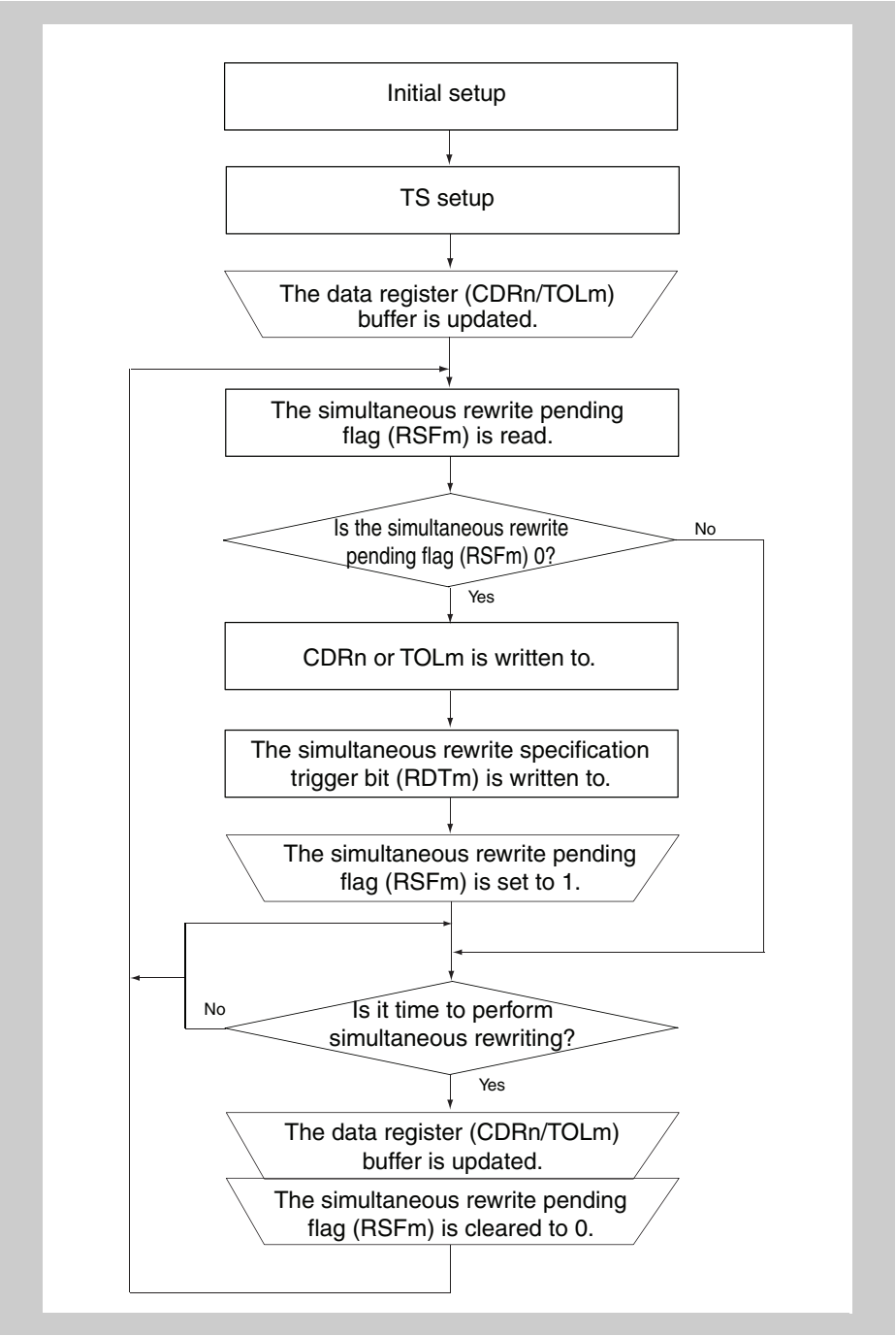


Figure 17-4 General procedure for simultaneous rewrite

(1) Initial settings

- To enable simultaneous rewrite in channel m, set TAUJnRDE.TAUJnRDEm = 1
- To select the type of simultaneous rewrite upon the start of counting by the master channel counter, specify the TAUJnRDM.TAUJnRDMm bits.

(2) Start counter and count operation

- To start all the TAUJnCNTm counters in the channel group, set the corresponding TAUJnTS.TAUJnTSm bits to 1. TAUJnTOL.TAUJnTOLm and the values in the data registers (TAUJnCDRm) are loaded to the corresponding TAUJnTOL.TAUJnTOLm buffer (TAUJnTOL.TAUJnTOLm buf) and data buffer registers (TAUJnCDRm buf) and the counters start.
- Setting the reload data trigger bit (TAUJnRDT.TAUJnRDTm) to 1 sets the reload flag (TAUJnRSF.TAUJnRSFm) to 1, enabling simultaneous rewrite. TAUJnRSF.TAUJnRSFm remains at 1 until simultaneous rewrite has taken place.
- When the specified trigger for simultaneous rewrite is detected, the TAUJnRSF.TAUJnRSFm bit is checked to see if simultaneous rewrite is enabled (TAUJnRSF.TAUJnRSFm = 1). If it is, simultaneous rewrite is carried out. If such writing is disabled, simultaneous rewriting is not performed, and the system awaits the detection of the next simultaneous rewrite trigger.

(3) Simultaneous rewrite

- When the simultaneous rewrite trigger is detected and simultaneous rewrite is enabled (TAUJnRSF.TAUJnRSFm = 1), the current values of the data registers are copied to their buffers. These values are then loaded to the corresponding counters and the values are applied the next time the counter starts or restarts.
- When simultaneous rewriting finishes, the TAUJnRSF.TAUJnRSFm bit is cleared to 0, and the system awaits the next simultaneous rewrite trigger.

17.8.3 Other general rules of simultaneous rewrite

The following rules also apply:

- TAUJnRDE.TAUJnRDEm and TAUJnRDM.TAUJnRDMm cannot be changed while the counter is in operation (TAUJnTE.TAUJnTEm = 1).
- TAUJnTOL.TAUJnTOLm can only be rewritten during operation when in PWM output function. For all other output functions, TAUJnTOL.TAUJnTOLm must be written before the counter starts. If it is rewritten in another function, TAUJnTTOUTm outputs an invalid wave.

17.8.4 Simultaneous rewrite procedure

Simultaneous rewrite is executed when the master channel starts or restarts counting.

The simultaneous rewrite procedure is described in the following figure.

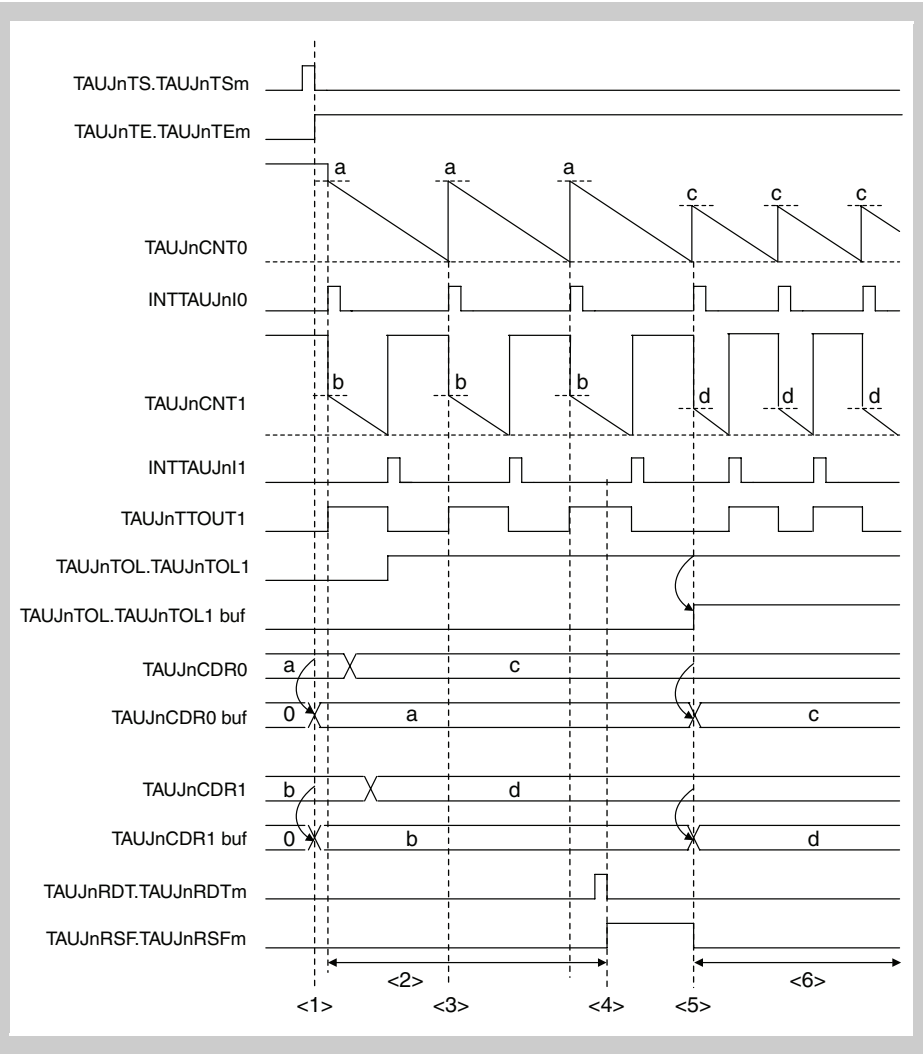


Figure 17-5 Simultaneous rewrite when the master channel (re)starts counting

Setup:

- CH0 is the master channel that counts down. CH1 represents an arbitrary slave channel. Simultaneous rewrite is executed when the master channel starts counting.

Description:

1. When TAUJnTS.TAUJnTSm is set to 1, the value of TAUJnCDRm is copied to the TAUJnCDRm buffer, and the value of TAUJnTOL.TAUJnTOLm is copied to the TAUJnTOL.TAUJnTOLm buffer.
2. The TAUJnCDRm and TAUJnTOL.TAUJnTOLm registers can be written at any time.
3. CH0 restarts counting, but simultaneous rewrite does not occur because it is disabled (TAUJnRSF.TAUJnRSFm = 0).
4. The reload data trigger bit (TAUJnRDT.TAUJnRDTm) is set to 1 which sets the status flag (TAUJnRSF.TAUJnRSFm = 1), enabling simultaneous rewrite.
5. Because simultaneous rewriting is enabled, it is performed before counting on channel 0 resumes. The TAUJnCDRm value is loaded to the TAUJnCDRm buffer, and the TAUJnTOL.TAUJnTOLm value is loaded to the TAUJnTOL.TAUJnTOLm buffer.
6. The counters count down and await the next simultaneous rewrite trigger. The values of TAUJnCDRm and TAUJnTOL.TAUJnTOLm can be changed again.

17.9 Channel output modes

The output of the TAUJnTTOUTm pin can be controlled in two ways, the latter of which can be further split into individual modes:

- By software (TAUJnTOE.TAUJnTOEm = 0)

  When controlled by software, the value written in the output register bit (TAUJnTO.TAUJnTOm) is output from the output pin (TAUJnTTOUTm).

- By TAUJ signals (TAUJnTOE.TAUJnTOEm = 1)

  When operated by TAUJ signals, the output level of TAUJnTTOUTm is set or reset or toggled by internal signals. The value of TAUJnTO.TAUJnTOm is updated accordingly to reflect the value of TAUJnTTOUTm.

  – Independently (TAUJnTOM.TAUJnTOMm = 0)

    When operated independently, the output of the TAUJnTTOUTm pin is only affected by settings of channel m. Therefore, independent channel operation must be specified (TAUJnTOM.TAUJnTOM = 0).

  – Synchronously (TAUJnTOM.TAUJnTOMm = 1)

    When operated synchronously, the output of the TAUJnTTOUTm pin is affected by settings of channel m and those of other channels. Therefore, synchronous channel operation must be selected for all participating channels (TAUJnTOM.TAUJnTOMm = 1).

The TAUJnTO.TAUJnTOm bit can always be read to determine the current value of TAUJnTTOUTm, regardless of whether the pin is controlled by software, operated independently, or operated synchronously.

Control bits

The settings of the control bits required to select a specific channel output mode are listed in *Table 17-11 “Channel output modes” on page 1189*.

The channel output modes are described in detail in

- *17.9.2 “Channel output modes controlled independently by TAUJn signals” on page 1191* to
- *17.9.3 “Channel output modes controlled synchronously by TAUJn signals” on page 1192*.

Collective TAUJnTOm bit manipulation

Whether to apply settings to the TAUJnTOm bits is controlled using the TAUJnTOE.TAUJnTOEm bits.

When writing to the TAUJnTO register, TAUJnTOm settings are only written to bits (channels) for which the corresponding TAUJnTOE.TAUJnTOEm bit is cleared to 0. The TAUJnTOm settings are not applied for bits (channels) for which the corresponding TAUJnTOE.TAUJnTOEm bit is set to 1.

Note

The TAUJnTO.TAUJnTOm bits are allocated so that the bit numbers correspond to the channel numbers.

Output logic

Positive logic or inverted logic of the output is specified by control bit TAUJnTOL.TAUJnTOLm.

The value of the TAUJnTOL.TAUJnTOLm bit must be set before the counter is started. It can only be changed during operation in PWM output function. If the TAUJnTOL.TAUJnTOLm bit is changed after counter operation starts, the TAUJnTTOUTm signal output becomes undefined.

See *17.8 “Simultaneous Rewrite” on page 1183*.

The various channel output modes and the channel output control bits when TAUJnTOC.TAUJnTOCm = 0 are listed in the following table.

Table 17-11 Channel output modes

Channel output mode	TAUJnTOE. TAUJnTOEm	TAUJnTOM. TAUJnTOMm
By software		
Independent channel output mode controlled by software	0	x
By TAUJ signals, independently		
Independent channel output mode 1	1	0
By TAUJ signals, synchronously		
Synchronous channel output mode 1	1	1

- The combinations not listed in this table are forbidden.
- The bit marked with an x can be set to any value.

Note The following bits cannot be changed during count operation (TAUJnTE.TAUJnTE = 1):

- TAUJnTOE.TAUJnTOEm
- TAUJnTOM.TAUJnTOMm
- TAUJnTOC.TAUJnTOCm

17.9.1 General procedure for specifying a channel output mode

The following steps describe the general procedure for specifying a TAUJnTTOUTm channel output mode. The prerequisite is that timer output operation is disabled (TAUJnTOE.TAUJnTOEm = 0).

1. Set TAUJnTO.TAUJnTOm to specify the initial level of the TAUJnTTOUTm output.
2. Set the channel output mode using *Table 17-11 “Channel output modes” on page 1189* and the output logic using the TAUJnTOL.TAUJnTOLm bit.
3. Start the counter (TAUJnTS.TAUJnTSm = 1).

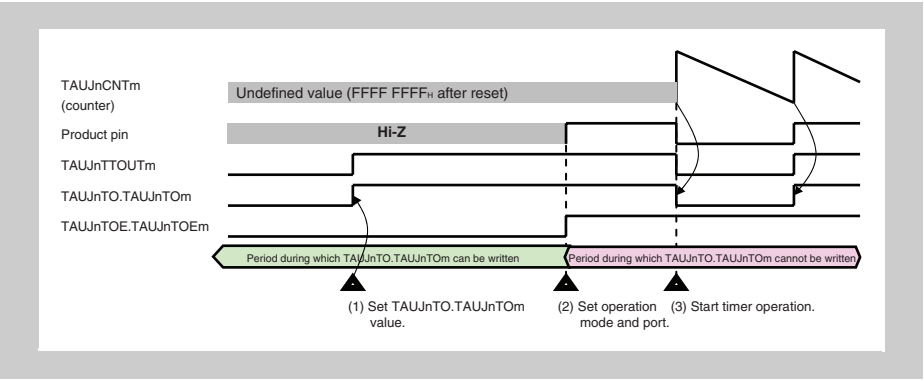


Figure 17-6 General procedure for specifying a TAUJnTTOUTm channel output mode

17.9.2 Channel output modes controlled independently by TAUJn signals

This section lists the channel output modes that are controlled independently by TAUJn signals. The control bits used to specify a mode are listed in *Table 17-11 “Channel output modes” on page 1189*.

(1) Independent channel output mode 1

Set/reset conditions In this output mode, TAUJnTTOUTm toggles when INTTAUJnIm is detected. The value of TAUJnTOL.TAUJnTOLm is ignored.

Prerequisites None, other than those in *Table 17-11 “Channel output modes” on page 1189*.

17.9.3 Channel output modes controlled synchronously by TAUJn signals

This section lists the channel output modes that are controlled synchronously by TAUJn signals. The control bits used to specify a mode are listed in *Table 17-11 “Channel output modes” on page 1189*.

(1) Synchronous channel output mode 1

Set/reset conditions In this output mode, INTTAUJnIm of the master channel serves as the set signal and INTTAUJnIm of the slave channel as the reset signal. If INTTAUJnIm of the master channel and INTTAUJnIm of the slave channel are generated at the same time, INTTAUJnIm of the slave channel (reset signal) has priority over INTTAUJnIm (set signal) of the master channel, i.e. the master channel is ignored.

Prerequisites None, other than those in *Table 17-11 “Channel output modes” on page 1189*.

17.10 Count Start Timing in Each Operating Mode

This section describes the timing for starting counting after the TAUJnTS.TAUJnTSm bit is set to 1, for each operation mode.

The value of the data register and whether an interrupt is generated depend on the individual mode and corresponding register settings.

Caution The timing for starting counting in this section is only for reference. The actual timing varies according to the count clock timing.

17.10.1 Interval timer mode and capture mode

The counter starts counting at the start of the next count clock cycle after TAUJnTS.TAUJnTSm is set to 1. The value of data register is also loaded when the counter starts.

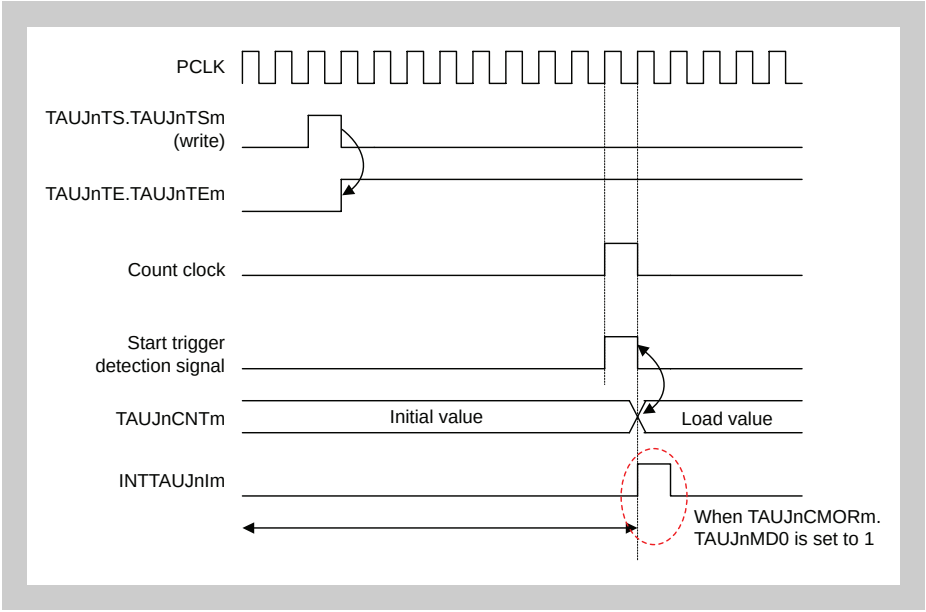


Figure 17-7 Count start timing in interval timer mode and capture mode

17.10.2 Other operating modes

In other operating modes, the count clock cycles are ignored with regard to starting the counter. The counter operation is only triggered by detection of a valid TAUJnTTINm edge. The value of data register is also loaded at the point the counter starts. Although the count clock cycle does not affect the timing for starting the counter operation, it determines the frequency with which all operations take place.

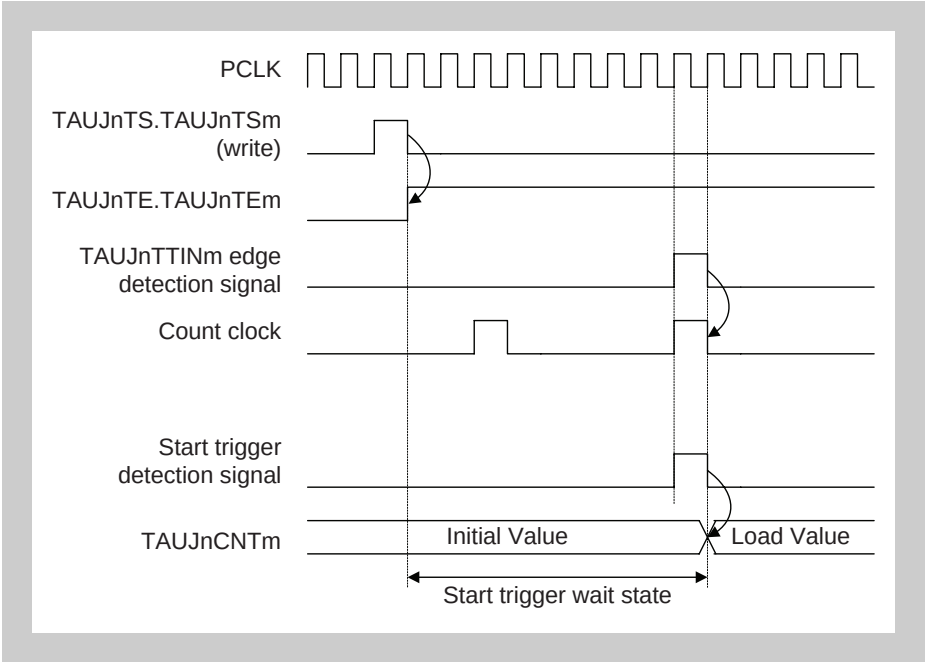


Figure 17-8 Count start timing in other operating modes

17.11 TAUJnTTOUTm Output and INTTAUJnIm Generation When Counter Starts or Restarts (by TAUJnMD0 Bit)

When the counter starts, it is possible to specify whether an INTTAUJnIm is generated using the TAUJnCMORm.TAUJnMD0 bit. The effect of the bit depends on the selected mode, as shown in the table below. The effects of INTTAUJnIm on TAUJnTTOUTm depend on the selected channel operation function.

Table 17-12 Effect of TAUJnCMORm.TAUJnMD0 bit on generation of INTTAUJnIm when counter is triggered

Mode	TAUJnCMORm. TAUJnMD0 bit	INTTAUJnIm generation upon start or restart of counting or upon trigger detection of TAUJnTTINm input signal
Interval timer mode Capture mode Count capture mode	0	No
	1	Yes
Capture & one-count mode Capture & gate count mode	0	No
One-count mode Gate count mode	0/1	No, regardless of the TAUJnCMOR.YAUJnMD0 bit setting

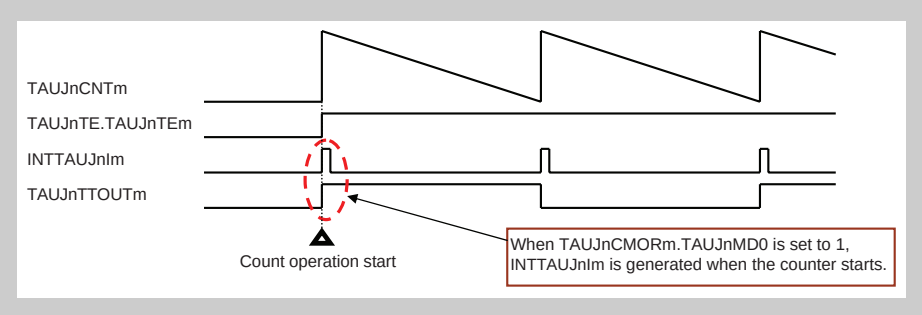


Figure 17-9 INTTAUJnIm generated when counter starts

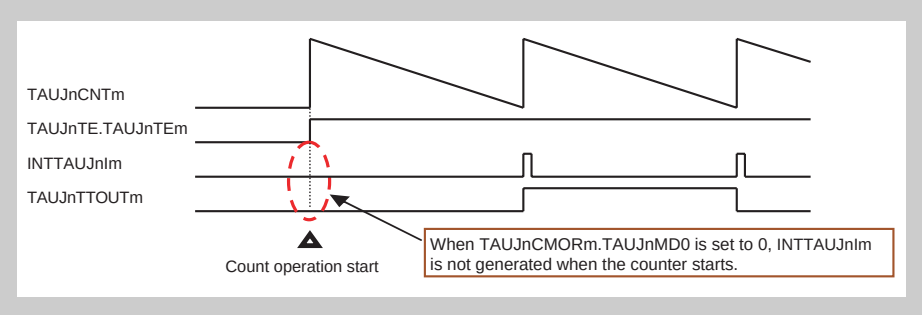


Figure 17-10 INTTAUJnIm not generated when counter starts

17.12 Interrupt Generation upon Overflow

For certain independent channel operation functions, no interrupt is generated even if the counter counts up to FFFF $FFFF_H$ and an overflow occurs. This section describes how it is possible to generate an interrupt, by combining a channel operating in one of these modes with a channel in a different operation mode which counts down.

The appropriate operation mode for the second channel depends on the operation mode of the first channel. Nevertheless, the principle is the same for all combinations:

- For the second channel, specify an operation mode in which the counter counts down and reaches 0000 0000_H at the same time that an overflow occurs on the first channel (TAUJnCNTm = FFFF $FFFF_H$).
- Set TAUJnCDRm of the second channel to FFFF $FFFF_H$
- The two channels must count at the same speed (i.e., they must have the same count clock).
- Both channels are triggered by the same TAUJnTTINm input
- The trigger detection settings (TAUJnCMORm.TAUJnSTS[2:0] and TAUJnCMURm.TAUJnTIS[1:0]) must be identical for both channels

Result: the down-counter of the second channel reaches 0000 0000_H at exactly the same time as the up-counter of the first channel overflows (TAUJnCNTm = FFFF $FFFF_H$). Thus the second channel generates the desired interrupt.

The following sections list the operating modes that count down that are required to match specific operating modes that count up, as well as example timing diagrams.

17.12.1 Capture mode

Applies to • TAUJnTTINm input pulse interval measurement function

Combine with Interval timer mode

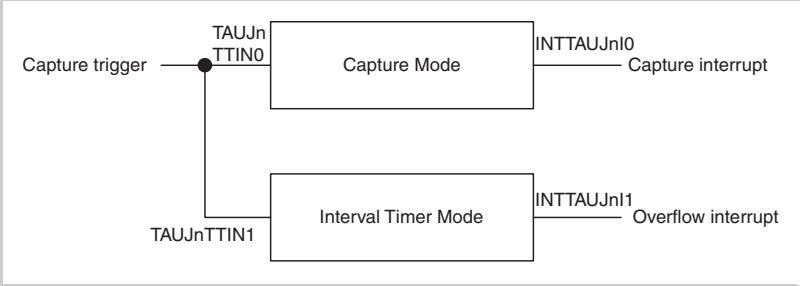


Figure 17-11 Combination of capture mode and interval timer mode

Timing diagram

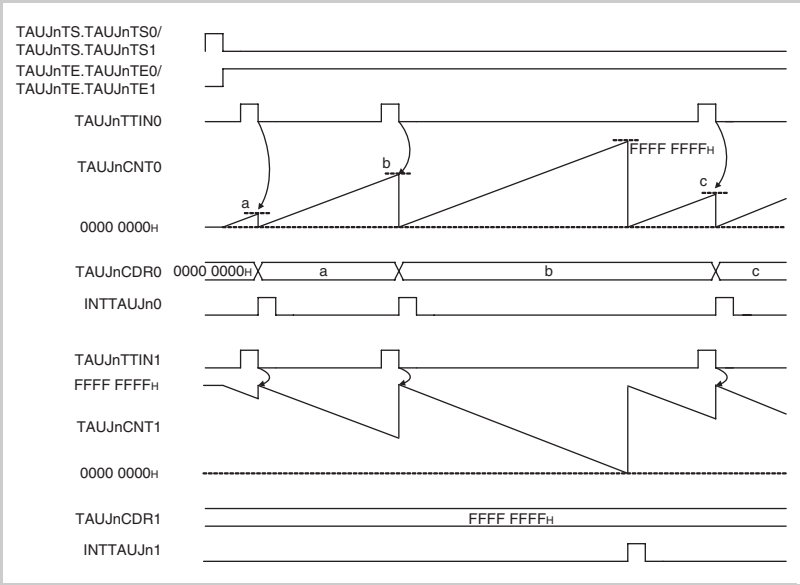


Figure 17-12 Interrupt generation via combination of capture mode and interval timer mode

17.12.2 Capture & one-count mode

Applies to
- TAUJnTTINm input signal width measurement function

Combine with
One-count mode

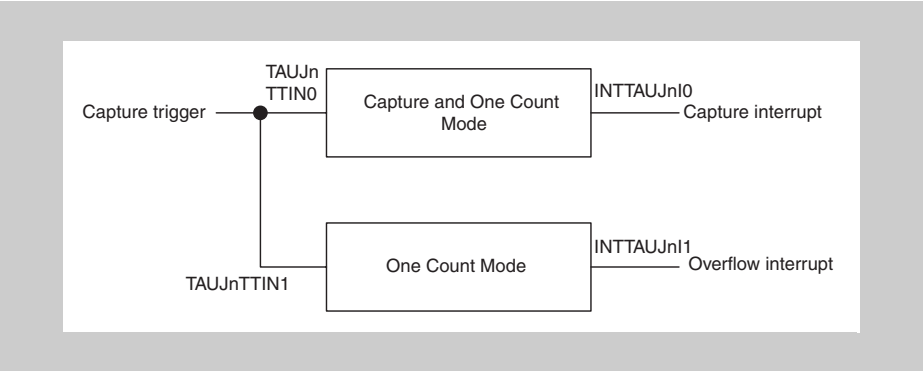


Figure 17-13 Combination of capture & one-count mode and one-count mode

Timing diagram

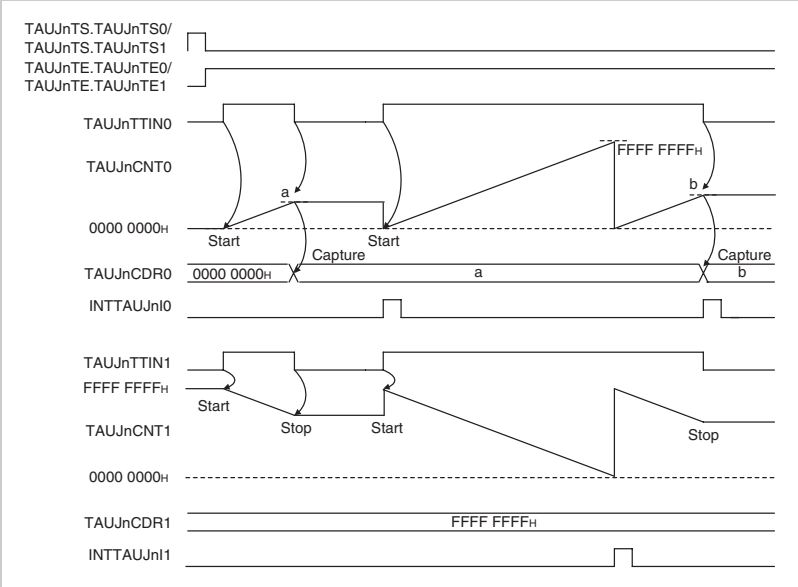


Figure 17-14 Interrupt generation via combination of capture & one-count mode and one-count mode

17.12.3 Count capture mode

Applies to
- TAUJnTTINm input position detection function

Combine with
Interval timer mode

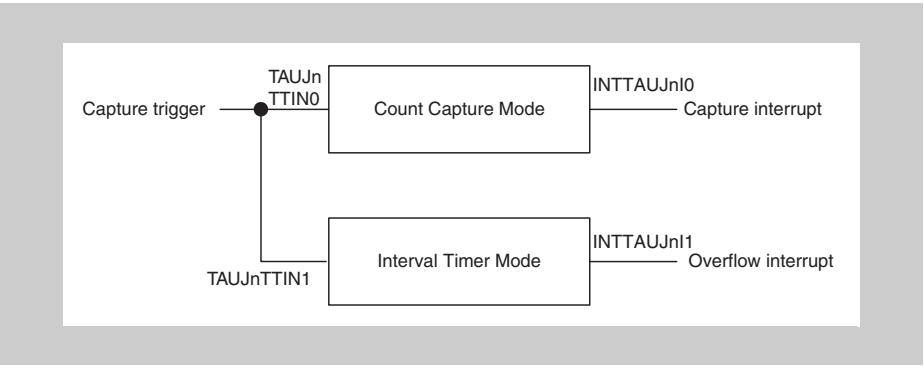


Figure 17-15 Combination of count capture mode and interval timer mode

Timing diagram

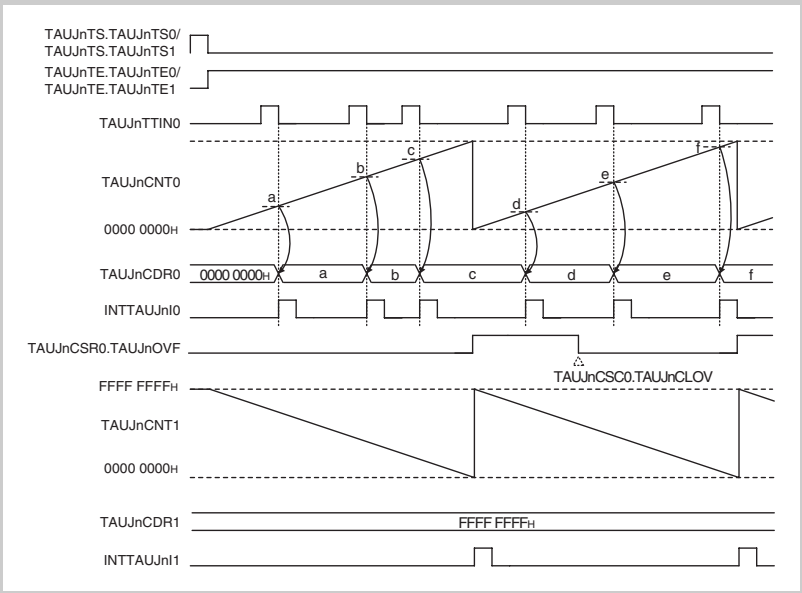


Figure 17-16 Interrupt generation via combination of count capture mode and interval timer mode

In the above timing diagram, TAUJnCSRm.TAUJnOVF is set to 1 when TAUJnCNTm overflows. TAUJnCSRm.TAUJnOVF is cleared by writing 1 to TAUJnCSCm.TAUJnCLOV.

17.12.4 Capture & gate count mode

Applies to
- TAUJnTTINm input period count detection function

Combine with
Gate count mode

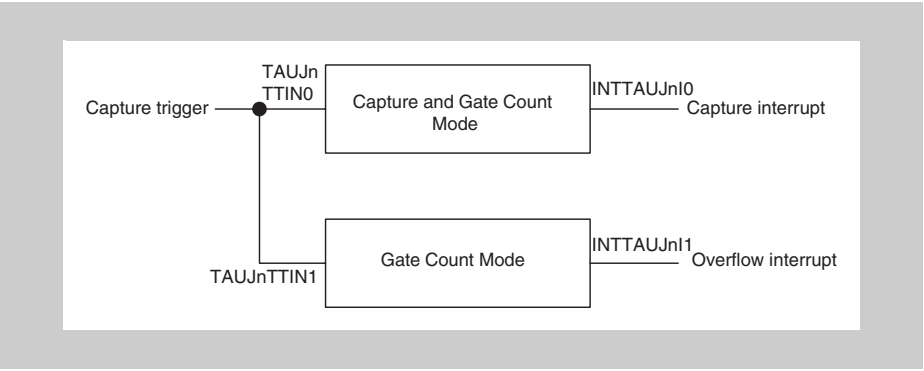


Figure 17-17 Combination of capture & gate count mode and gate count mode

Timing diagram

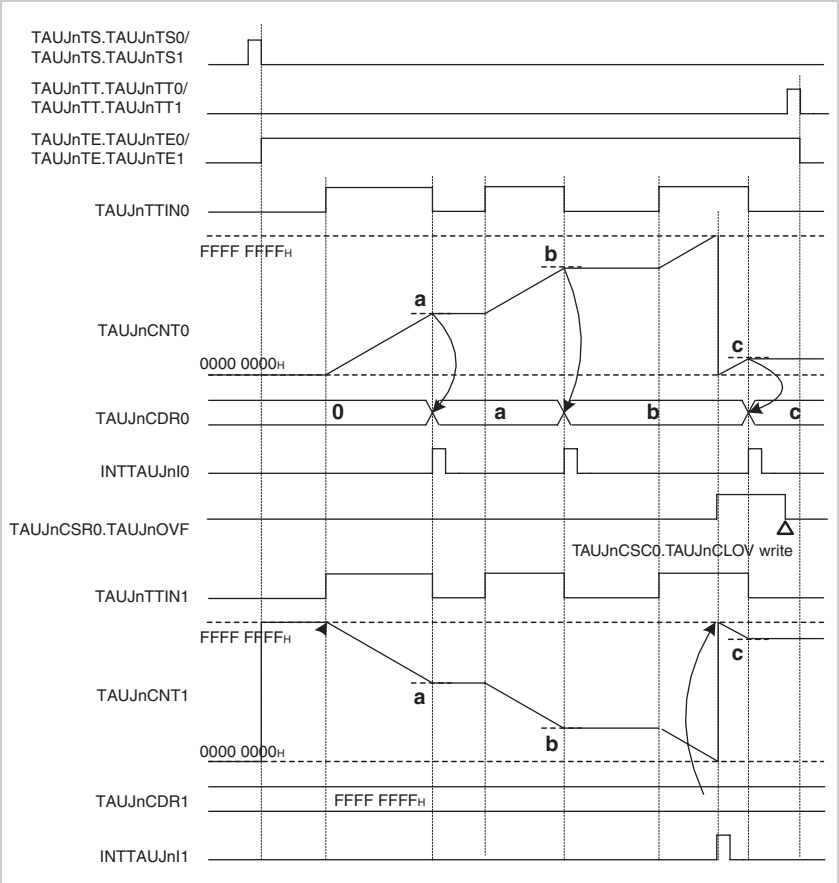


Figure 17-18 Interrupt generation via combination of capture & gate count mode and gate count mode

In the above timing diagram, TAUJnCSRm.TAUJnOVF is set to 1 when TAUJnCNTm overflows. TAUJnCSRm.TAUJnOVF is cleared by writing 1 to TAUJnCSCm.TAUJnCLOV.

17.13 TAUJnTTINm Edge Detection

Edge detection is based on the operation clock. This means that an edge can only be detected at the next rising edge of the operation clock. This can lead to a maximum delay of one operation clock cycle.

The following figure shows an edge detection timing example.

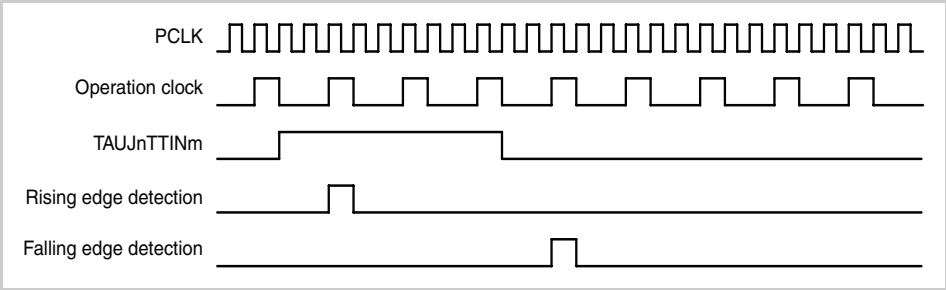


Figure 17-19 Basic edge detection timing

Figure 17-19 "Basic edge detection timing" shows the approximate operation timing. Actually, there is delay due to the noise filter and synchronizer between the TAUJnIm pin and TAUJn.

17.14 Independent Channel Operation Functions

The following sections list the independent channel operation functions provided by TAUJ. For a general overview of independent channel operation, see *17.4 “Functional Description” on page 1177*.

17.15 Independent Channel Interrupt Functions

This section describes functions that generate interrupts at regular intervals.

- *17.15.1 “Interval timer function”*
- *17.15.2 “TAUJnTTINm input interval timer function”*

17.15.1 Interval timer function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUJnIm) at regular intervals. When an interrupt is generated, the TAUJnTTOUTm signal toggles, resulting in a square wave.

Prerequisites
- The operation mode must be set to interval timer mode, see *Table 17-13 "TAUJnCMORm settings for interval timer function" on page 1205*.
- The channel output mode must be set to independent channel output mode 1, see *17.9 "Channel output modes" on page 1188*.

Description The counter operation is enabled by setting the channel trigger bit (TAUJnTS.TAUJnTSm) to 1. This in turn sets TAUJnTE.TAUJnTEm = 1, enabling count operation. The current value of TAUJnCDRm is loaded to TAUJnCNTm and the counter starts counting down from this value.

When the counter reaches 0000 0000_H, INTTAUJnIm is generated and the TAUJnTTOUTm signal toggles. Next, TAUJnCNTm loads the value of TAUJnCDRm, and then subsequently continues operation.

The value of TAUJnCDRm can be rewritten at any time, and the changed value of TAUJnCDRm is applied the next time the counter starts counting down.

The counter can be stopped by setting TAUJnTT.TAUJnTTm to 1, which in turn sets TAUJnTE.TAUJnTEm to 0. TAUJnCNTm and TAUJnTTOUTm stop but retain their values. The counter can be reset by setting TAUJnTS.TAUJnTSm to 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUJnTS.TAUJnTSm to 1 during operation.

Conditions If the TAUJnCMORm.TAUJnMD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUJnTTOUTm does not toggle. This results in an inverted TAUJnTTOUTm signal compared to when TAUJnCMORm.TAUJnMD0 is set to 1. For details, see *17.11 "TAUJnTTOUTm Output and INTTAUJnIm Generation When Counter Starts or Restarts (by TAUJnMD0 Bit)" on page 1195*.

(2) Equations

INTTAUJnIm cycle = count clock cycle x (TAUJnCDRm + 1)

TAUJnTTOUTm square wave cycle = count clock cycle x (TAUJnCDRm + 1) x 2

(3) Block diagram and general timing diagram

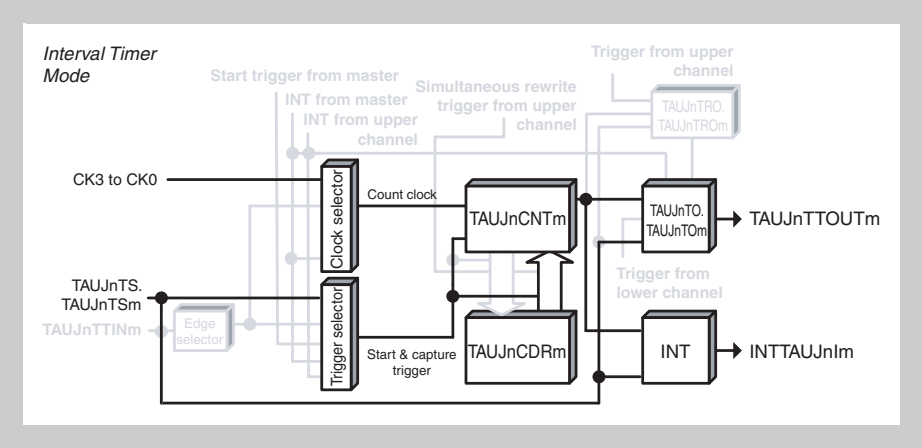


Figure 17-20 Block diagram for interval timer function

The following settings apply to the general timing diagram:

- INTTAUJnIm is generated at operation start (TAUJnCMORm.TAUJnMD0 = 1)

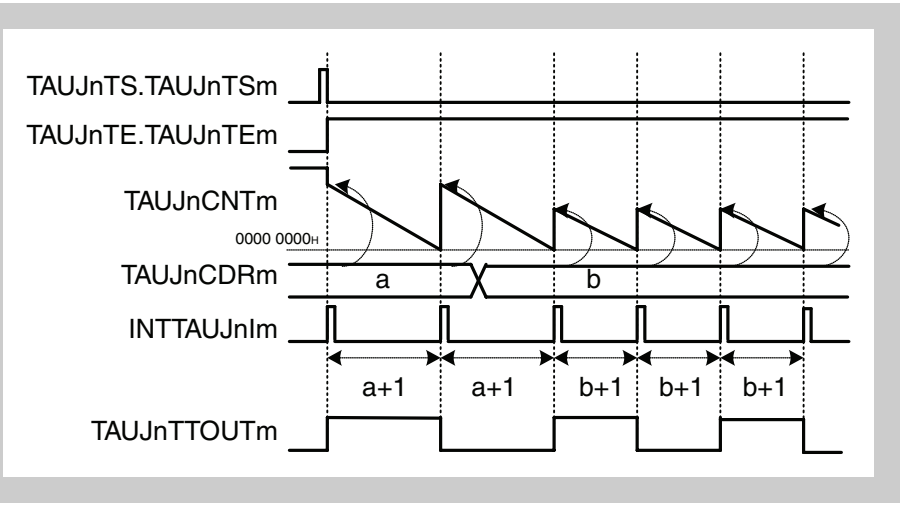


Figure 17-21 General timing diagram for interval timer function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-13 TAUJnCMORm settings for interval timer function

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output = CK0 01: Prescaler output = CK1 10: Prescaler output = CK2 11: Prescaler output = CK3
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	0: Not used, so set to 0
TAUJnSTS[2:0]	000: Counter triggered by software trigger
TAUJnCOS[1:0]	00: Not used, so set to 00
TAUJnMD[4:1]	0000: Interval timer mode
TAUJnMD0	0: INTTAUJnIm not generated and TAUJnTTOUTm does not toggle at operation start or restart 1: Generates INTTAUJnIm and toggles TAUJnTTOUTm at operation start or restart

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-14 TAUJnCMURm settings for interval timer function

Bit name	Setting
TAUJnTIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Table 17-15 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUJnTOE.TAUJnTOEm	1: Disables independent channel output mode controlled by software
TAUJnTOM.TAUJnTOMm	0: Independent channel output
TAUJnTOC.TAUJnTOCm	0: Operation mode 1 (= Toggle mode if TAUJnTOM.TAUJnTOMm = 0)
TAUJnTOL.TAUJnTOLm	0: Positive logic

Note The channel output mode can also be set to channel output mode controlled by software by setting TAUJnTOE.TAUJnTOEm = 0. TAUJnTTOUTm can then be controlled independently of the interrupts. For details, see *17.9 "Channel output modes" on page 1188*.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the interval timer function. Therefore, these registers must be set to 0.

Table 17-16 Simultaneous rewrite settings for interval timer function

Bit name	Setting
TAUJnRDE.TAUJnRDEm	0: Disables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.TAUJnRDEm = 0), set these bits to 0

(5) Operating procedure for interval timer function

Table 17-17 Operating procedure for interval timer function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm and TAUJnCMURm registers as described in *Table 17-13 "TAUJnCMORm settings for interval timer function" on page 1205* and *Table 17-14 "TAUJnCMURm settings for interval timer function" on page 1205*. Set the value of the TAUJnCDRm register. Set the channel output mode by setting the control bits as described in *Table 17-15 "Control bit settings for independent channel output mode 1" on page 1206*.	Channel operation is stopped.
Start operation	Set TAUJnTS.TAUJnTSm to 1. TAUJnTS.TAUJnTSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is set to 1 and the counter starts. TAUJnCNTm loads the TAUJnCDRm value. When TAUJnCMORm.TAUJnMD0 = 1, INTTAUJnIm is generated and TAUJnTTOUTm toggles.
During operation	The TAUJnCDRm register value can be changed at any time. The TAUJnCNTm register can be read at all times.	TAUJnCNTm counts down. When the counter reaches 0000_H: • TAUJnCNTm reloads the TAUJnCDRm value, and then continues count operation. • INTTAUJnIm is generated and TAUJnTTOUTm toggles.
Stop operation	Set TAUJnTT.TAUJnTTm to 1. TAUJnTT.TAUJnTTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is cleared to 0 and the counter stops. TAUJnCNTm and TAUJnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

(a) TAUJnCDRm = 0000 0000$_H$, count clock = PCLK/2

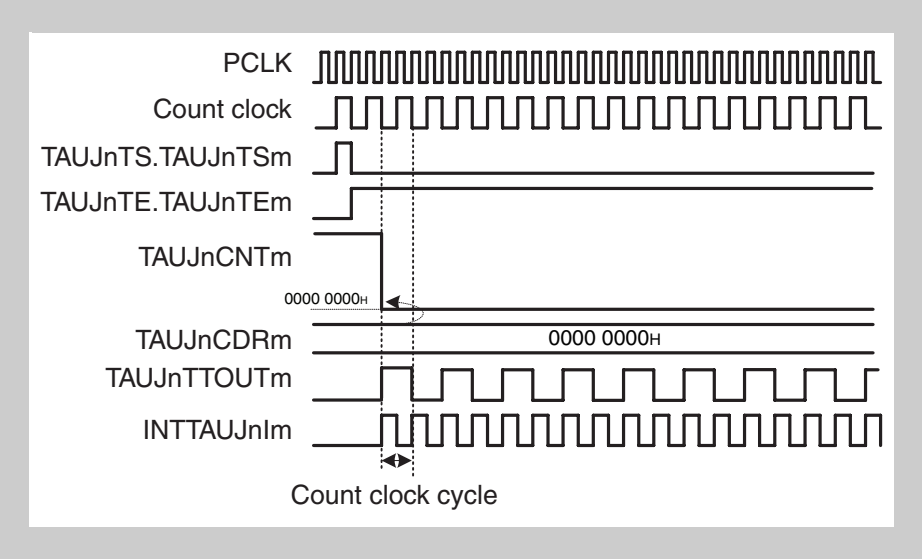


Figure 17-22 TAUJnCDRm = 0000 0000$_H$, count clock = PCLK/2

- If TAUJnCDRm = 0000 0000$_H$ and the count clock = PCLK/2[1], the TAUJnCDRm value is loaded to TAUJnCNTm every count clock, meaning that TAUJnCNTm is always 0000 0000$_H$.
- INTTAUJnIm is generated every count clock, resulting in TAUJnTTOUTm toggling every count clock.

(b) TAUJnCDRm = 0000 0000$_H$, count clock = PCLK

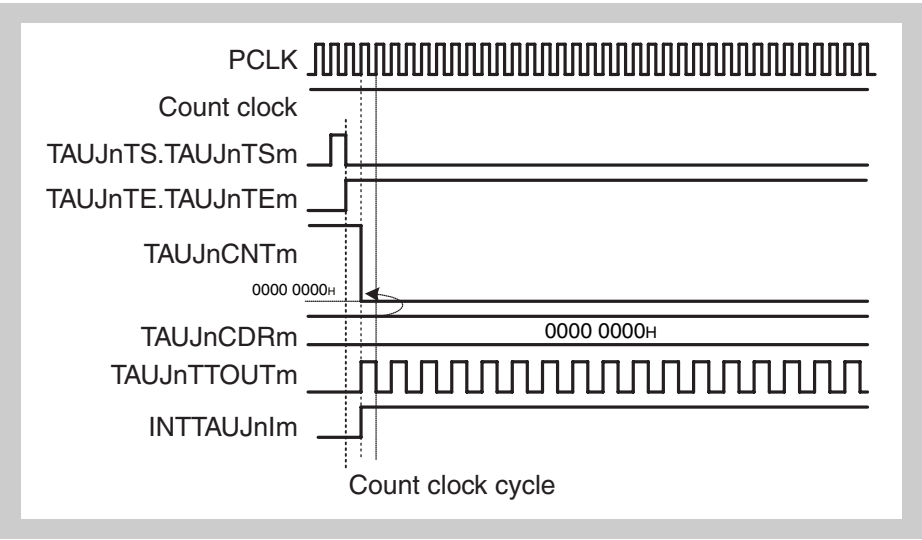


Figure 17-23 TAUJnCDRm = 0000 0000$_H$, count clock = PCLK

- If TAUJnCDRm = 0000 0000$_H$ and the count clock = PCLK, the TAUJnCDRm value is loaded to TAUJnCNTm every PCLK clock, meaning that TAUJnCNTm is always 0000 0000$_H$.
- INTTAUJnIm is generated continuously, resulting in TAUJnTTOUTm toggling every PCLK clock.

(c) Operation stop and restart

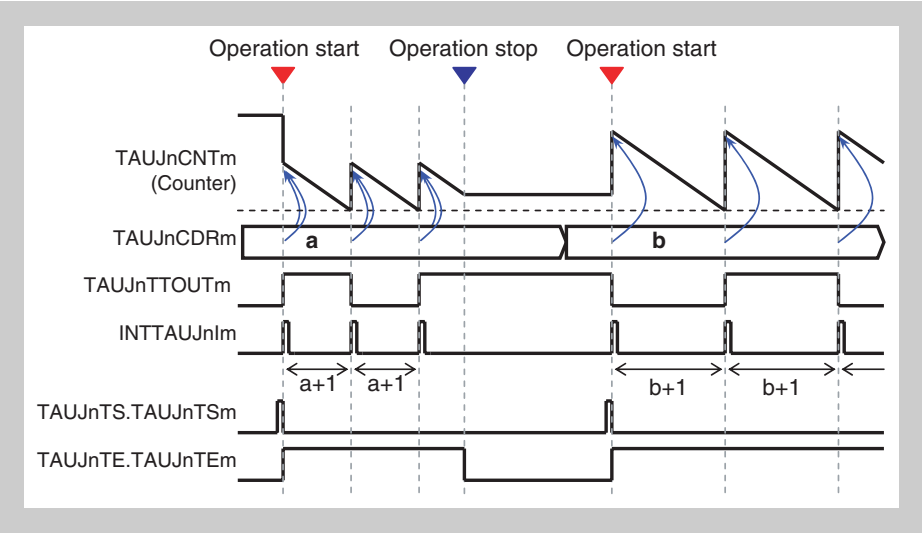


Figure 17-24 Operation stop and restart, TAUJnCMORm.TAUJnMD0 = 1

- The counter can be stopped by setting TAUJnTT.TAUJnTTm to 1, which in turn sets TAUJnTE.TAUJnTEm to 0.
- TAUJnCNTm and TAUJnTTOUTm stop but retain their values.
- The counter can be restarted by setting TAUJnTS.TAUJnTSm to 1.

(d) Forced restart

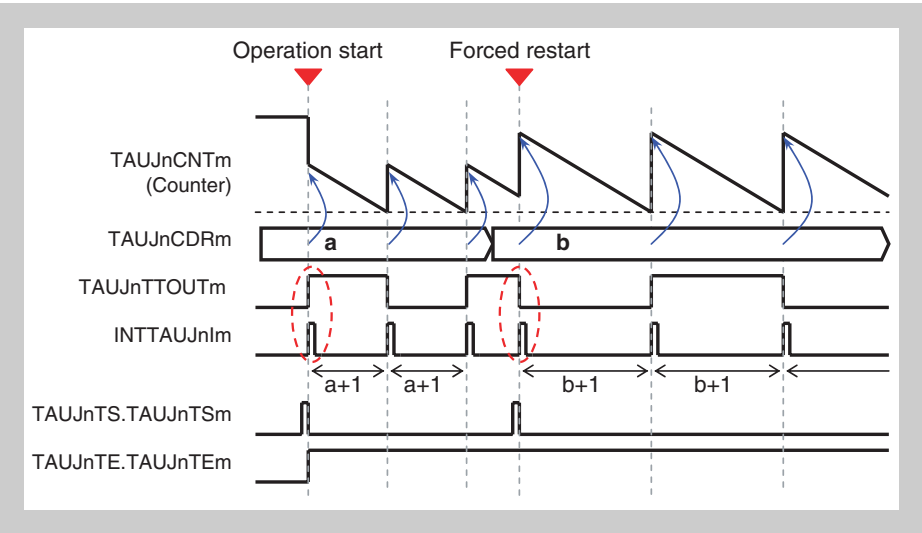


Figure 17-25 Forced restart operation, TAUJnCMORm.TAUJnMD0 = 1

- The counter can be forcibly restarted (without stopping it first) by setting TAUJnTS.TAUJnTSm to 1 during operation.
- If the TAUJnCMORm.TAUJnMD0 bit is set to 1, the first interrupt after a start or restart is generated.

17.15.2 TAUJnTTINm input interval timer function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUJnIm) at regular intervals or when a valid TAUJnTTINm input edge is detected. When an interrupt is generated, the TAUJnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to interval timer mode, see *Table 17-18 "TAUJnCMORm settings for TAUJnTTINm input interval timer function" on page 1212*.
- The channel output mode must be set to independent channel output mode 1, see *17.9 "Channel output modes" on page 1188*.

Description This function operates in an identical manner to the interval timer function (see *17.15.1 "Interval timer function" on page 1203*), except that this function is restarted by a valid TAUJnTTINm input edge. The type of edge used as the trigger is specified using the TAUJnCMURm.TAUJnTIS[1:0] bits. Either rising edge, falling edge, or rising and falling edge can be selected.

(2) Equations

INTTAUJnIm cycle = count clock cycle x (TAUJnCDRm + 1)

TAUJnTTOUTm square wave cycle = count clock cycle x (TAUJnCDRm + 1) x 2

(3) Block diagram and general timing diagram

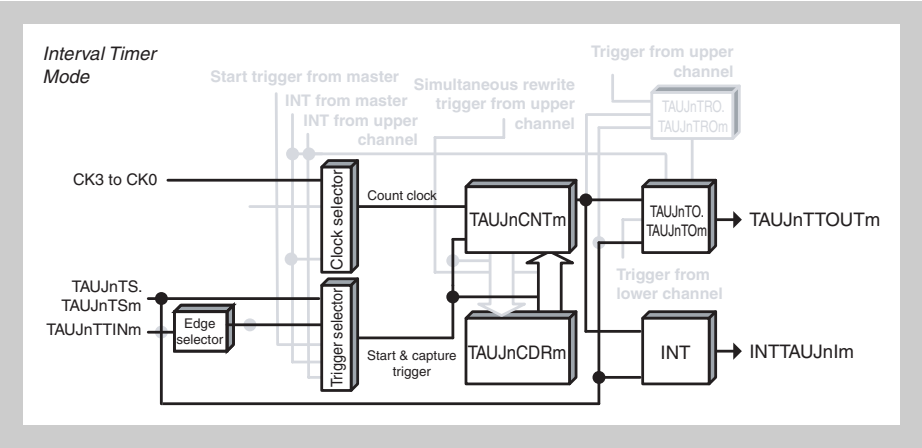


Figure 17-26 Block diagram for TAUJnTTINm input interval timer function

The following settings apply to the general timing diagram:

- INTTAUJnIm is generated at operation start (TAUJnCMORm.TAUJnMD0 = 1).
- Rising edge detection (TAUJnCMURm.TAUJnTIS[1:0] = 01_B)

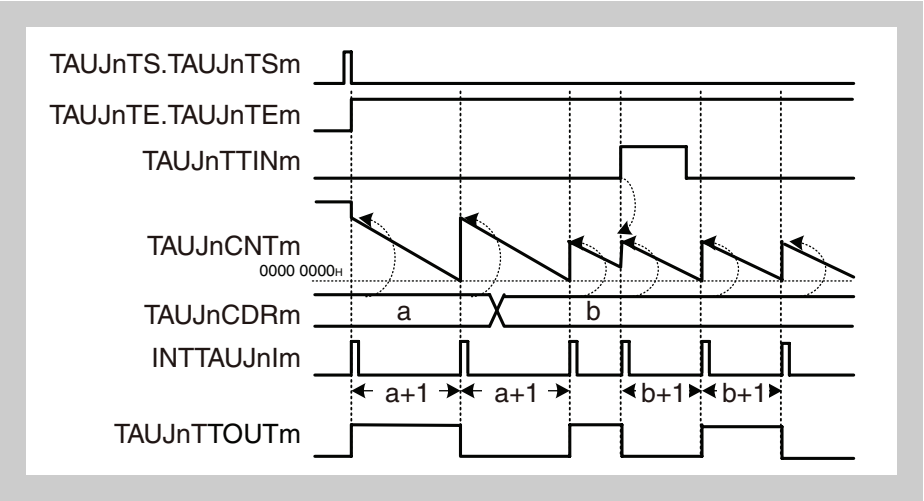


Figure 17-27 General timing diagram for TAUJnTTINm input interval timer function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-18 TAUJnCMORm settings for TAUJnTTINm input interval timer function

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output= CK0 01: Prescaler output = CK1 10: Prescaler output = CK2 11: Prescaler output = CK3
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	0: Not used, so set to 0
TAUJnSTS[2:0]	001: Valid TAUJnTTINm input edge signal is used as the external start trigger
TAUJnCOS[1:0]	00: Not used, so set to 00
TAUJnMD[4:1]	0000: Interval timer mode
TAUJnMD0	0: INTTAUJnIm not generated and TAUJnTTOUTm does not toggle at operation start 1: Generates INTTAUJnIm and toggles TAUJnTTOUTm at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-19 TAUJnCMURm settings for TAUJnTTINm input interval timer function

Bit name	Setting
TAUJnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

Table 17-20 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUJnTOE.TAUJnTOEm	1: Enables independent channel output mode controlled by software
TAUJnTO.TAUJnTOm	0: Low level 1: High level
TAUJnTOM.TAUJnTOMm	0: Independent channel output
TAUJnTOC.TAUJnTOCm	0: Operation mode 1 (= Toggle mode if TAUJnTOM.TAUJnTOMm = 0)
TAUJnTOL.TAUJnTOLm	0: Positive logic

Note The channel output mode can also be set to channel output mode controlled by software by setting TAUJnTOE.TAUJnTOEm = 0. TAUJnTTOUTm can then be controlled independently of the interrupts. For details, see *17.9 "Channel output modes" on page 1188*.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm input interval timer function. Therefore, these registers must be set to 0.

Table 17-21 Simultaneous rewrite settings for TAUJnTTINm input interval timer function

Bit name	Setting
TAUJnRDE.TAUJnRDEm	0: Disables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.TAUJnRDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm input interval timer function

Table 17-22 Operating procedure for TAUJnTTINm input interval timer function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 17-18 "TAUJnCMORm settings for TAUJnTTINm input interval timer function" on page 1212* and *Table 17-19 "TAUJnCMURm settings for TAUJnTTINm input interval timer function" on page 1212*. Set the value of the TAUJnCDRm register. Set the channel output mode by setting the control bits as described in *Table 17-20 "Control bit settings for independent channel output mode 1" on page 1213*.	Channel operation is stopped.
Start operation	Set TAUJnTS.TAUJnTSm to 1. TAUJnTS.TAUJnTSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is set to 1 and the counter starts. TAUJnCNTm loads the TAUJnCDRm value. When TAUJnCMORm.TAUJnMD0 = 1, INTTAUJnIm is generated and TAUJnTTOUTm toggles.
During operation	The values of the TAUJnCMURm.TAUJnTIS[1:0] and the TAUJnCDRm register can be changed at any time. The TAUJnCNTm register can be read at all times. Detection of TAUJnTTINm edge	TAUJnCNTm counts down. When the counter reaches 0000 0000_H: • TAUJnCNTm reloads the TAUJnCDRm value and continues count operation. • INTTAUJnIm is generated and TAUJnTTOUTm toggles. When a TAUJnTTINm input valid edge is detected during count operation, TAUJnCNTm reloads the TAUJnCDRm value and continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TAUJnTTm to 1. TAUJnTT.TAUJnTTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is cleared to 0 and the counter stops. TAUJnCNTm and TAUJnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

The timing diagrams in *17.15.1 "Interval timer function" on page 1203* also apply, except for this function the counter can also be restarted by a valid TAUJnTTINm input edge.

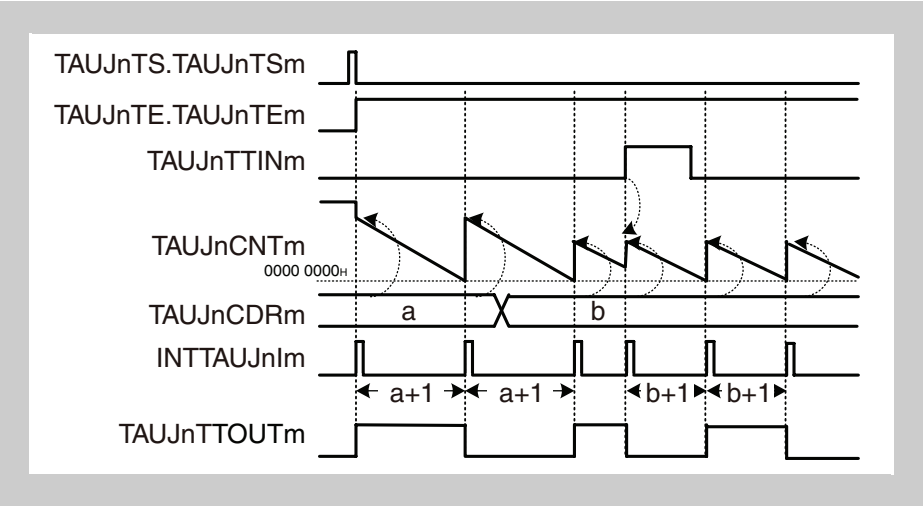


Figure 17-28 Counter triggered by rising TAUJnTTINm input edge (TAUJnCMURm.TAUJnTIS[1:0] = 01_B), TAUJnCMORm.TAUJnMD0 = 1

- If a valid TAUJnTTINm input edge is detected, an interrupt is generated which causes TAUJnTTOUTm to toggle. In this example, the valid edge is a rising edge (TAUJnCMURm.TAUJnTIS[1:0] = 01_B).

17.16 Independent Channel Signal Measurement Functions

This section describes functions that measure the widths of an individual TAUJnTTINm pulse or the total width of successive TAUJnTTINm pulses. It also describes functions that measure the interval of the signal or that compare the width of a pulse with a reference value.

- *17.16.1 “TAUJnTTINm input pulse interval measurement function”*
- *17.16.2 “TAUJnTTINm input signal width measurement function”*
- *17.16.3 “Overflow interrupt output function (during TAUJnTTINm width measurement)”*
- *17.16.4 “TAUJnTTINm input period count detection function”*
- *17.16.5 “Overflow interrupt output function (during TAUJnTTINm input period count detection)”*

17.16.1 TAUJnTTINm input pulse interval measurement function

(1) Overview

Summary This function captures the count value and uses this value and the overflow bit TAUJnCSRm.TAUJnOVF to measure the interval of the TAUJnTTINm input signal.

Prerequisites

- The operation mode must be set to capture mode, see *Table 17-24 "TAUJnCMORm settings for TAUJnTTINm input pulse interval measurement function" on page 1219*.
- TAUJnTTOUTm is not used for this function.

Description The counter operation is enabled by setting the channel trigger bit (TAUJnTS.TAUJnTSm) to 1. This in turn sets TAUJnTE.TAUJnTEm = 1, enabling count operation. The counter TAUJnCNTm starts counting up from 0000 0000_H. When a valid TAUJnTTINm edge is detected, the value of TAUJnCNTm is captured, transferred to TAUJnCDRm, and an interrupt INTTAUJnIm is generated. The counter resets to 0000 0000_H and subsequently continues operation.

If the counter reaches FFFF $FFFF_H$ before a valid TAUJnTTINm edge is detected, it overflows to 0000 0000_H. The counter is reset to 0000 0000_H and subsequently continues operation. The values transferred to TAUJnCDRm and TAUJnCSRm.TAUJnOVF respectively depend on the values of bits TAUJnCMORm.TAUJnCOS[1:0]:

Table 17-23 Effects of an overflow

TAUJnCMORm.COS[1:0]	When overflow occurs		When a valid TAUJnTTINm input is then detected	
	TAUJnCDRm	TAUJnCSRm.TAUJnOVF	TAUJnCDRm and TAUJnCNTm	TAUJnCSRm.TAUJnOVF
00	Unchanged	0	TAUJnCNTm loaded to TAUJnCDRm	1
01		1		
10	Set to FFFF $FFFF_H$	0	TAUJnCNTm set to 0, TAUJnCDRm unchanged	0
11		1		

When TAUJnCMORm.TAUJnCOS[0] is 1, the overflow bit TAUJnCSRm.TAUJnOVF can only be cleared by setting TAUJnCSCm.TAUJnCLOV to 1.

The combination of the value of TAUJnCDRm and TAUJnCSRm.TAUJnOVF can be used to deduce the interval of the TAUJnTTINm signal. However, if an overflow occurs multiple times before a valid TAUJnTTINm input is detected, the overflow bit TAUJnCSRm.TAUJnOVF cannot indicate this.

The function can be stopped by setting TAUJnTT.TAUJnTTm = 1, which in turn sets TAUJnTE.TAUJnTEm = 0. TAUJnCNTm stops and retains its value. While the function is stopped, TAUJnTTINm input valid edge detection and TAUJnCNTm capture are not performed.

The counter is reset to 0000 0000_H and subsequently continues operation. The counter can also be forcibly restarted (without stopping it first) by setting TAUJnTS.TAUJnTSm = 1 during operation.

Conditions If the TAUJnCMORm.TAUJnMD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details, see *17.11 "TAUJnTTOUTm Output and INTTAUJnIm Generation When Counter Starts or Restarts (by TAUJnMD0 Bit)" on page 1195*.

Note When TAUJnCMORm.TAUJnCOS[1:0] = 11_B, the value of TAUJnCNTm is *not* loaded to TAUJnCDRm when the first valid TAUJnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUJnTTINm input pulse interval = count clock cycle x [(TAUJnCSRm.TAUJnOVF x (FFFF $FFFF_H$ + 1)) + TAUJnCDRm capture value + 1]

(3) Block diagram and general timing diagram

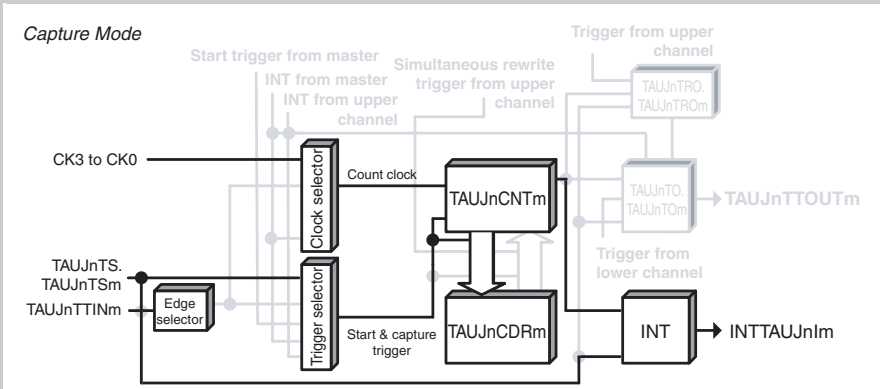


Figure 17-29 Block diagram for TAUJnTTINm input pulse interval measurement function

The following settings apply to the general timing diagram:

- INTTAUJnIm not generated at operation start (TAUJnCMORm.TAUJnMD0 = 0)
- Falling edge detection (TAUJnCMURm.TAUJnTIS[1:0] = 00_B)
- When a valid TAUJnTTINm input is detected after an overflow TAUJnCDRm is changed and TAUJnCSRm.TAUJnOVF is set to 1 (TAUJnCMORm.TAUJnCOS[1:0] = 00_B)

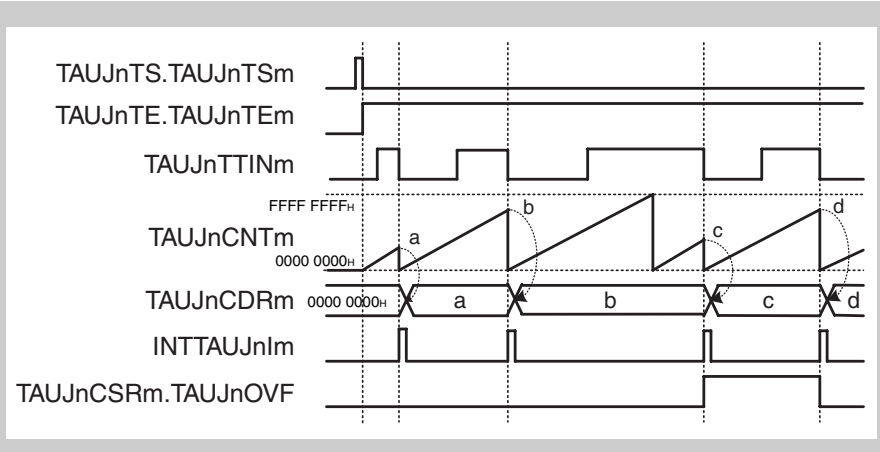


Figure 17-30 General timing diagram for TAUJnTTINm input pulse interval measurement function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-24 TAUJnCMORm settings for TAUJnTTINm input pulse interval measurement function

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output = CK0 01: Prescaler output = CK1 10: Prescaler output = CK2 11: Prescaler output = CK3
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	0: Not used, so set to 0
TAUJnSTS[2:0]	001: Valid edge of the TAUJnTTINm input signal is the external capture trigger
TAUJnCOS[1:0]	See *Table 17-23 “Effects of an overflow” on page 1217*
TAUJnMD[4:1]	0010: Capture mode
TAUJnMD0	0: INTTAUJnIm not generated at operation start 1: Generates INTTAUJnIm at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-25 TAUJnCMURm settings for TAUJnTTINm input pulse interval measurement function

Bit name	Setting
TAUJnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm input pulse interval measurement function. Therefore, these registers must be set to 0.

Table 17-26 Simultaneous rewrite settings for TAUJnTTINm input pulse interval measurement function

Bit name	Setting
TAUJnRDE.TAUJnRDEm	0: Disables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.TAUJnRDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm input pulse interval measurement function

Table 17-27 Operating procedure for TAUJnTTINm input pulse interval measurement function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 17-24 "TAUJnCMORm settings for TAUJnTTINm input pulse interval measurement function" on page 1219* and *Table 17-25 "TAUJnCMURm settings for TAUJnTTINm input pulse interval measurement function" on page 1219*. Set the value of the TAUJnCDRm register.	Channel operation is stopped.
Start operation	Set TAUJnTS.TAUJnTSm to 1. TAUJnTS.TAUJnTSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is set to 1 and the counter starts. TAUJnCNTm is cleared to 0000 0000_H. INTTAUJnIm is generated when TAUJnCMORm.TAUJnMD0 is set to 1.
During operation	Detection of TAUJnTTINm edges. The TAUJnCMURm.TAUJnTIS[1:0] bits can be changed at any time. The TAUJnCDRm and TAUJnCSRm registers can be read at any time. Writing 1 to the TAUJnCSCm.TAUJnCLOV bit is possible. (The TAUJnCSRm.TAUJnOVF bit is cleared to 0.)	TAUJnCNTm starts counting up from 0000 0000_H. When a TAUJnTTINm valid edge is detected: • TAUJnCNTm transfers (captures) its value to TAUJnCDRm, and returns to 0000 0000_H. • INTTAUJnIm is then generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TAUJnTTm to 1. TAUJnTT.TAUJnTTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is cleared to 0 and the counter stops. TAUJnCNTm stops and both it and TAUJnCSRm.TAUJnOVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUJnCMORm.TAUJnCOS[1:0] = 00_B

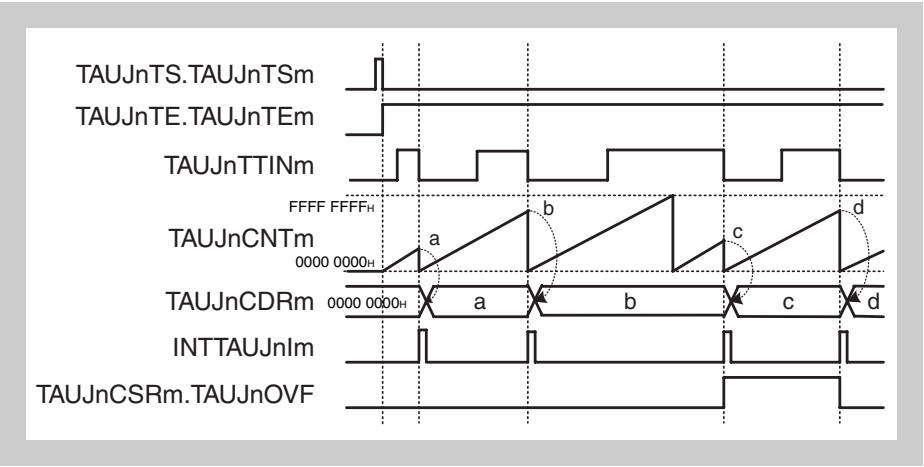


Figure 17-31 TAUJnCMORm.TAUJnCOS[1:0] = 00_B, TAUJnCMORm.TAUJnMD0 = 0, TAUJnCMURm.TAUJnTIS[1:0] = 00_B

- When an overflow occurs, the value of TAUJnCDRm remains unchanged and TAUJnCSRm.TAUJnOVF remains 0.
- Upon detection of the next valid TAUJnTTINm input edge, the value of TAUJnCNTm is loaded to TAUJnCDRm and TAUJnCSRm.TAUJnOVF is set to 1.
- Upon detecting the next valid TAUJnTTINm input edge while no overflow has occurred, TAUJnCSRm.TAUJnOVF is cleared to 0.

(b) TAUJnCMORm.TAUJnCOS[1:0] = 01_B

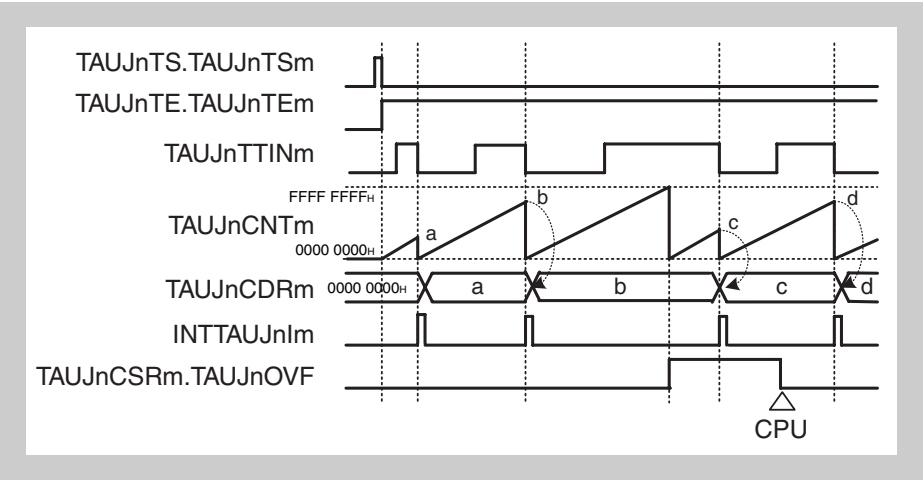


Figure 17-32 TAUJnCMORm.TAUJnCOS[1:0] = 01_B, TAUJnCMORm.TAUJnMD0 = 0, TAUJnCMURm.TAUJnTIS[1:0] = 00_B

- When an overflow occurs, the value of TAUJnCDRm remains unchanged and TAUJnCSRm.TAUJnOVF is set to 1.
- Upon detection of the next valid TAUJnTTINm input edge, the value of TAUJnCNTm is loaded to TAUJnCDRm.
- TAUJnCSRm.TAUJnOVF is only cleared by a CPU command (setting the TAUJnCSCm.TAUJnCLOV bit to 1).

(c) TAUJnCMORm.TAUJnCOS[1:0] = 10_B

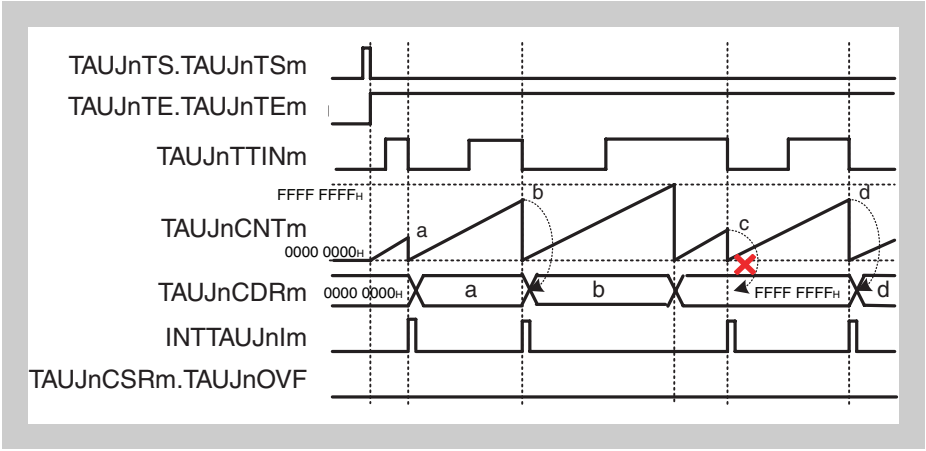


Figure 17-33 TAUJnCMORm.TAUJnCOS[1:0] = 10_B, TAUJnCMORm.TAUJnMD0 = 0, TAUJnCMURm.TAUJnTIS[1:0] = 00_B

- When an overflow occurs, TAUJnCDRm is set to FFFF FFFF$_H$ and TAUJnCSRm.TAUJnOVF remains 0.
- Upon detection of the next valid TAUJnTTINm input edge, TAUJnCNTm is reset to 0, but TAUJnCDRm and TAUJnCSRm.TAUJnOVF remain unchanged.
- Thus, the next TAUJnTTINm input valid edge after the overflow is ignored.

(d) TAUJnCMORm.TAUJnCOS[1:0] = 11_B

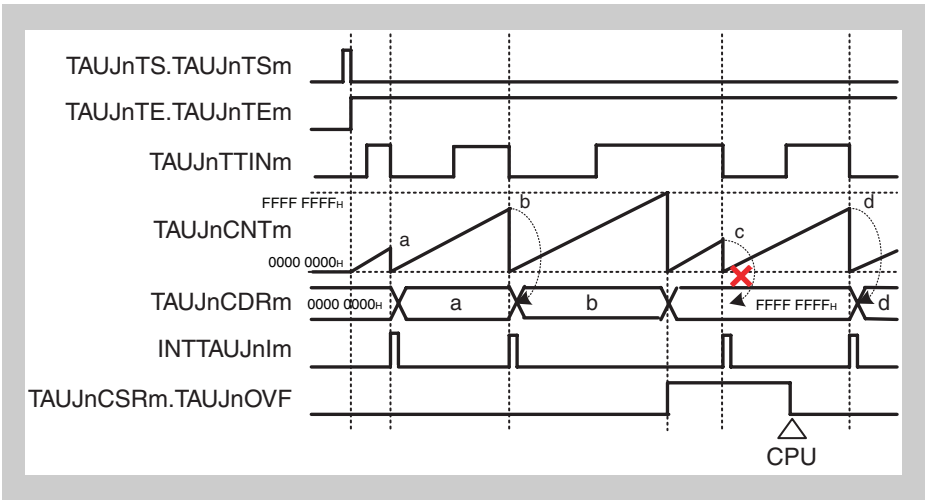


Figure 17-34 TAUJnCMORm.TAUJnCOS[1:0] = 11_B, TAUJnCMORm.TAUJnMD0 = 0, TAUJnCMURm.TAUJnTIS[1:0] = 00_B

- When an overflow occurs, TAUJnCDRm is set to FFFF FFFF$_H$, and TAUJnCSRm.TAUJnOVF is set to 1.
- Upon detection of the next valid TAUJnTTINm input edge, TAUJnCNTm is reset to 0, but TAUJnCDRm and TAUJnCSRm.TAUJnOVF remain unchanged.
- Thus, the next TAUJnTTINm input valid edge after the overflow is ignored.
- TAUJnCSRm.TAUJnOVF is cleared by setting TAUJnCSCm.TAUJnCLOV to 1.

17.16.2 TAUJnTTINm input signal width measurement function

(1) Overview

Summary This function measures the width of a TAUJnTTINm input signal.

Prerequisites

- The operation mode must be set to Capture & one-count mode, see *Table 17-29 "TAUJnCMORm settings for TAUJnTTINm input signal width measurement function" on page 1226*.
- TAUJnTTOUTm is not used for this function.
- TAUJnCMORm.TAUJnMD0 must be set to 0.

Description

<R> The counter operation is enabled by setting the channel trigger bit (TAUJnTS.TAUJnTSm) to 1. This in turn sets TAUJnTE.TAUJnTEm = 1, enabling count operation. When a valid TAUJnTTINm start edge is detected, the counter TAUJnCNTm starts counting up from 0000 0000_H. When a valid TAUJnTTINm stop edge is detected, the value of TAUJnCNTm is captured, transferred to TAUJnCDRm, and an interrupt INTTAUJnIm is generated. The counter retains its value (TAUJnCDRm + 1) and awaits the next valid TAUJnTTINm input start edge.

If the counter reaches FFFF $FFFF_H$ before a valid TAUJnTTINm stop edge is detected, it overflows. The counter is reset to 0000 0000_H and subsequently continues operation. The values transferred to TAUJnCDRm and TAUJnCSRm.TAUJnOVF respectively depend on the values of bits TAUJnCMORm.TAUJnCOS[1:0]:

Table 17-28 Effects of an overflow

TAUJnCMORm. COS[1:0]	When overflow occurs		When a valid TAUJnTTINm input stop edge is detected	
	TAUJnCDRm	TAUJnCSRm. TAUJnOVF	TAUJnCDRm and TAUJnCNTm	TAUJnCSRm. TAUJnOVF
00	Unchanged	0	TAUJnCNTm loaded to TAUJnCDRm	1
01		1		
10	Set to FFFF $FFFF_H$	0	TAUJnCNTm stops counting TAUJnCDRm unchanged	0
11		1		

When TAUJnCMORm.TAUJnCOS[0] is 1, the overflow bit TAUJnCSRm.TAUJnOVF can only be cleared by setting TAUJnCSCm.TAUJnCLOV to 1.

The combination of the value of TAUJnCDRm and TAUJnCSRm.TAUJnOVF can be used to deduce the width of the TAUJnTTINm signal. However, if an overflow occurs multiple times before a valid TAUJnTTINm input is detected, the overflow bit TAUJnCSRm.TAUJnOVF cannot indicate this.

This function cannot be forcibly restarted.

Note When TAUJnCMORm.TAUJnCOS[1] = 1, the value of TAUJnCNTm is not loaded to TAUJnCDRm when the first valid TAUJnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUJnTTINm input signal width = count clock cycle x [(TAUJnCSRm.TAUJnOVF x (FFFF $FFFF_H$ + 1)) + TAUJnCDRm capture value + 1]

(3) Block diagram and general timing diagram

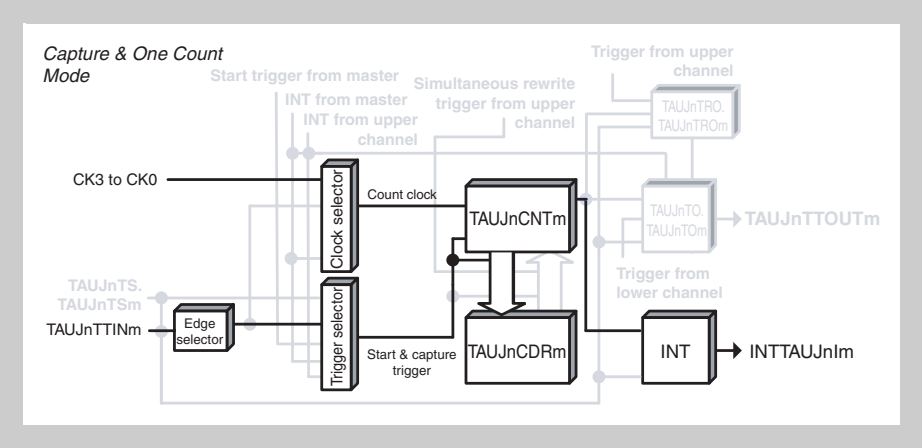


Figure 17-35 Block diagram for TAUJnTTINm input signal width measurement function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUJnCMURm.TAUJnTIS[1:0] = 11_B)
- When a valid TAUJnTTINm input is detected after an overflow TAUJnCDRm is changed and TAUJnCSRm.TAUJnOVF is set to 1 (TAUJnCMORm.TAUJnCOS[1:0] = 00_B)

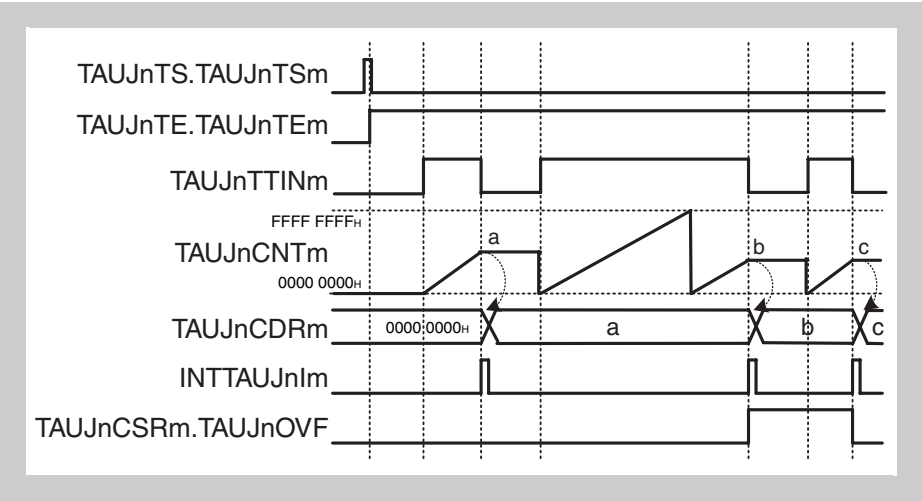


Figure 17-36 General timing diagram for TAUJnTTINm input signal width measurement function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-29 TAUJnCMORm settings for TAUJnTTINm input signal width measurement function

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output = CK0 01: Prescaler output = CK1 10: Prescaler output = CK2 11: Prescaler output = CK3
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	0: Not used, so set to 0
TAUJnSTS[2:0]	010: Valid edge of the TAUJnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
TAUJnCOS[1:0]	See *Table 17-28 “Effects of an overflow” on page 1224*.
TAUJnMD[4:1]	0110: Capture & one-count mode
TAUJnMD0	0: Disables the start trigger during operation

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-30 TAUJnCMURm settings for TAUJnTTINm input signal width measurement function

Bit name	Setting
TAUJnTIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

Because the channel output mode is not used for this function, clear TAUJnTOE.TAUJnTOEm to 0. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm input signal width measurement function. Therefore, these registers must be set to 0.

Table 17-31 Simultaneous rewrite settings for TAUJnTTINm input signal width measurement function

Bit name	Setting
TAUJnRDE.TAUJnRDEm	0: Disables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.TAUJnRDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm input signal width measurement function

Table 17-32 Operating procedure for TAUJnTTINm input signal width measurement function

<R>

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 17-24 "TAUJnCMORm settings for TAUJnTTINm input pulse interval measurement function" on page 1219* and *Table 17-25 "TAUJnCMURm settings for TAUJnTTINm input pulse interval measurement function" on page 1219*. Set the value of the TAUJnCDRm register.	Channel operation is stopped.
Start operation	Set TAUJnTS.TAUJnTSm to 1. TAUJnTS.TAUJnTSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is set to 1 and TAUJnCNTm waits for detection of the TAUJnTTINm start edge. When a TAUJnTTINm start is detected, TAUJnCNTm starts counting up.
During operation	Detection of TAUJnTTINm edges. The TAUJnCDRm, TAUJnCNTm, and TAUJnCSRm registers can be read at any time. The TAUJnCSCm.CLOV bit can be set to 1.	TAUJnCNTm starts counting up from 0000 0000_H. When a TAUJnTTINm valid edge is detected: • TAUJnCNTm transfers (captures) its value to TAUJnCDRm, and retains its value INTTAUJnIm is then generated. • Counting stops at the "value transferred to TAUJnCDRm + 1" value and TAUJnCNTm waits for detection of the TAUJnTTINm start edge. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TAUJnTTm to 1. TAUJnTT.TAUJnTTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is cleared to 0 and the counter stops. TAUJnCNTm stops and both it and TAUJnCSRm.TAUJnOVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUJnCMORm.TAUJnCOS[1:0] = 00_B

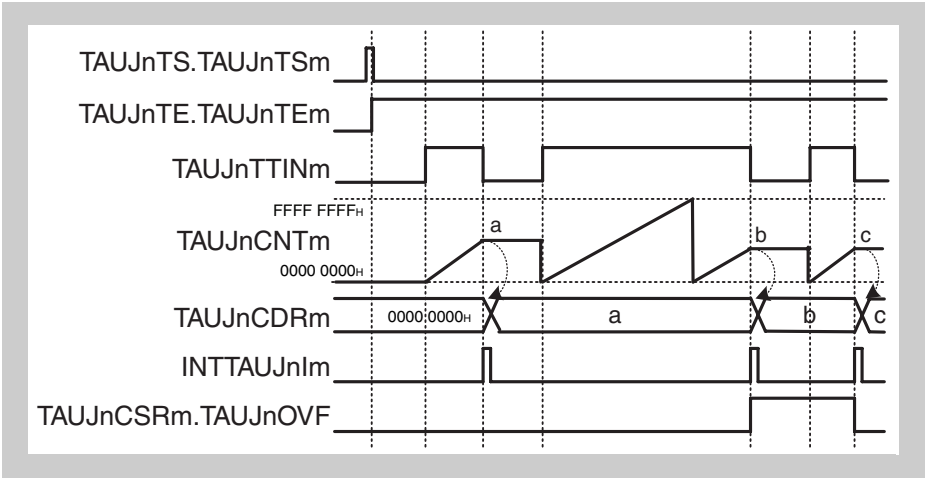


Figure 17-37 TAUJnCMORm.TAUJnCOS[1:0] = 00_B, TAUJnCMORm.TAUJnMD0 = 0, TAUJnCMURm.TAUJnTIS[1:0] = 11_B

- When an overflow occurs, the value of TAUJnCDRm remains unchanged and TAUJnCSRm.TAUJnOVF remains 0.
- Upon detection of the next valid TAUJnTTINm input edge, the value of TAUJnCNTm is loaded to TAUJnCDRm and TAUJnCSRm.TAUJnOVF is set.
- Upon detecting the next valid TAUJnTTINm input edge while no overflow has occurred, TAUJnCSRm.TAUJnOVF is cleared to 0.

(b) TAUJnCMORm.TAUJnCOS[1:0] = 01_B

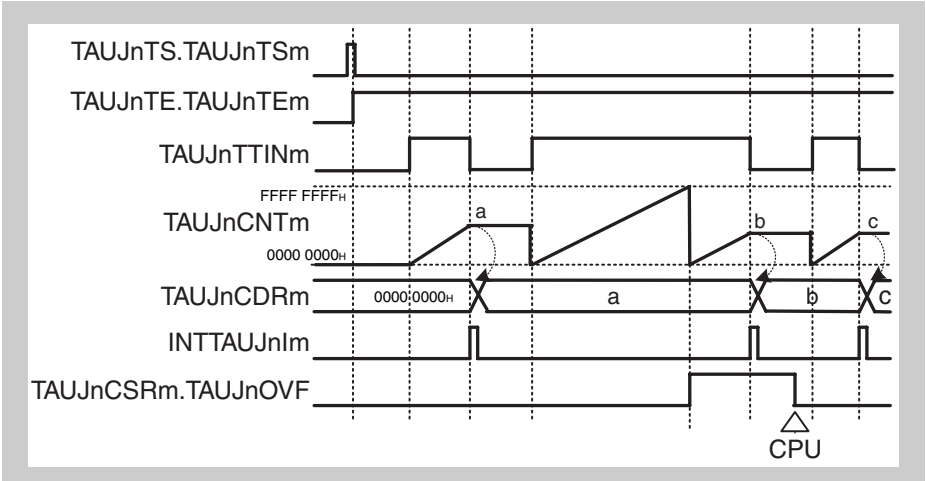


Figure 17-38 TAUJnCMORm.TAUJnCOS[1:0] = 01_B, TAUJnCMORm.TAUJnMD0 = 0, TAUJnCMURm.TAUJnTIS[1:0] = 11_B

- When an overflow occurs, the value of TAUJnCDRm remains unchanged and TAUJnCSRm.TAUJnOVF is set to 1.
- Upon detection of the next valid TAUJnTTINm input edge, the value of TAUJnCNTm is loaded to TAUJnCDRm.
- TAUJnCSRm.TAUJnOVF is only cleared by a CPU command (setting the TAUJnCSCm.TAUJnCLOV bit to 1).

(c) TAUJnCMORm.TAUJnCOS[1:0] = 10_B

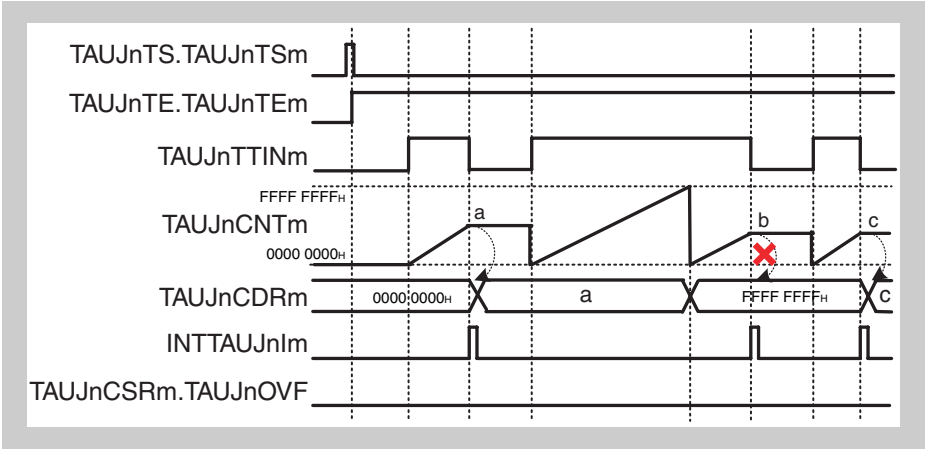


Figure 17-39 TAUJnCMORm.TAUJnCOS[1:0] = 10_B, TAUJnCMORm.TAUJnMD0 = 0, TAUJnCMURm.TAUJnTIS[1:0] = 11_B

- When an overflow occurs, TAUJnCDRm is set to FFFF FFFF$_H$ and TAUJnCSRm.TAUJnOVF remains 0.
- Upon detection of the next valid TAUJnTTINm input edge, TAUJnCNTm is reset to 0, but TAUJnCDRm and TAUJnCSRm.TAUJnOVF remain unchanged.
- Thus, the next TAUJnTTINm input valid edge after the overflow is ignored.

(d) TAUJnCMORm.TAUJnCOS[1:0] = 11_B

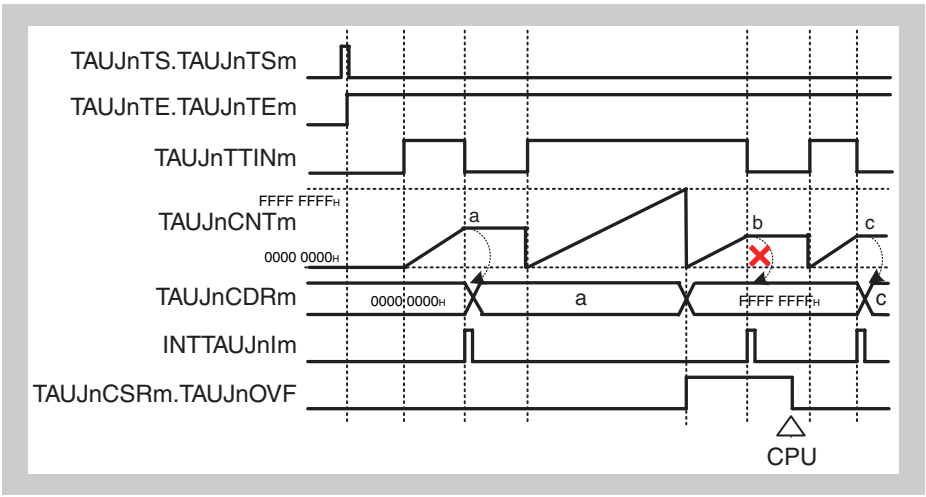


Figure 17-40 TAUJnCMORm.TAUJnCOS[1:0] = 11_B, TAUJnCMORm.TAUJnMD0 = 0, TAUJnCMURm.TAUJnTIS[1:0] = 11_B

- When an overflow occurs, TAUJnCDRm is set to FFFF FFFF$_H$, and TAUJnCSRm.TAUJnOVF is set to 1.
- Upon detection of the next valid TAUJnTTINm input edge, TAUJnCNTm is reset to 0, but TAUJnCDRm and TAUJnCSRm.TAUJnOVF remain unchanged.
- Thus, the next TAUJnTTINm input valid edge after the overflow is ignored.
- TAUJnCSRm.TAUJnOVF is cleared by setting TAUJnCSCm.TAUJnCLOV to 1.

17.16.3 Overflow interrupt output function (during TAUJnTTINm width measurement)

(1) Overview

Summary This function measures the width of an individual TAUJnTTINm input signal. An interrupt is generated if (FFFF $FFFF_H$ + 1) is exceeded following TAUJnTTINm input.

Prerequisites
- The operation mode must be set to one-count mode, see *Table 17-33 "TAUJnCMORm settings for overflow interrupt output function (during TAUJnTTINm width measurement)" on page 1233*.
- TAUJnTTOUTm is not used for this function.
- The value of TAUJnCDRm must be set to FFFF $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUJnTS.TAUJnTSm) to 1. This in turn sets TAUJnTE.TAUJnTEm = 1, enabling count operation.

The counter starts when a valid TAUJnTTINm input start edge is detected. FFFF $FFFF_H$ is loaded to TAUJnCNTm and the counter starts counting down.

When a valid stop edge is detected, the counter stops and retains the current value.

When the next TAUJnTTINm input start edge is detected, TAUJnCNTm loads FFFF $FFFF_H$ and starts counting down.

If the counter reaches 0000 0000_H before a stop edge is detected, an interrupt is generated.

Conditions The valid start and stop edges are specified by the TAUJnCMURm.TAUJnTIS[1:0] bits:
- If TAUJnCMURm.TAUJnTIS[1:0] = 10_B, the TAUJnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUJnCMURm.TAUJnTIS[1:0] = 11_B, the TAUJnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

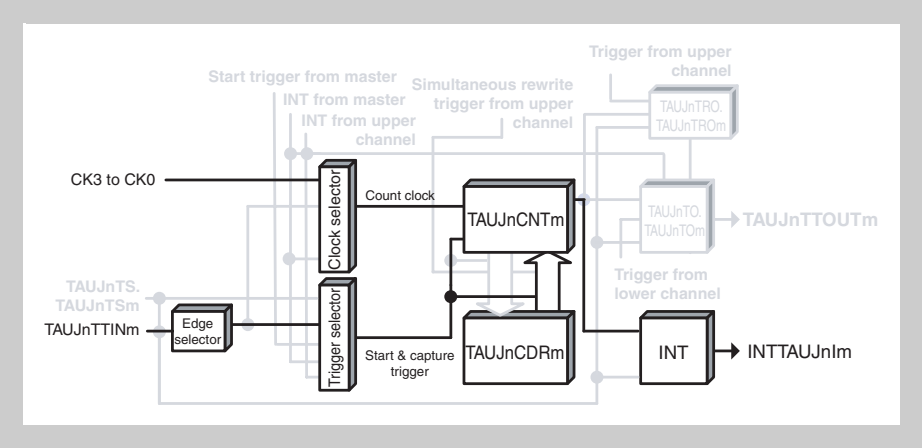


Figure 17-41 Block diagram for overflow interrupt output function (during TAUJnTTINm width measurement)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUJnCMURm.TAUJnTIS[1:0] = 11_B)

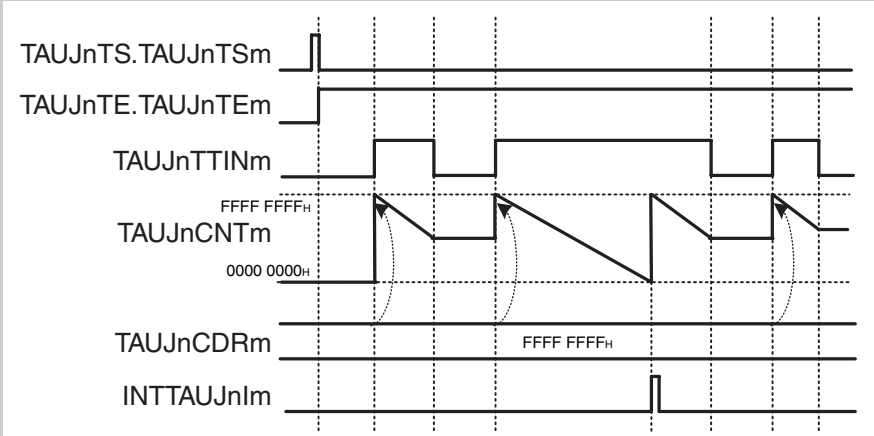


Figure 17-42 General timing diagram for overflow interrupt output function (during TAUJnTTINm width measurement)

(3) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-33 TAUJnCMORm settings for overflow interrupt output function (during TAUJnTTINm width measurement)

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output = CK0 01: Prescaler output = CK1 10: Prescaler output = CK2 11: Prescaler output = CK3
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	0: Not used, so set to 0
TAUJnSTS[2:0]	010: Valid edge of the TAUJnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
TAUJnCOS[1:0]	00: Not used, so set to 00
TAUJnMD[4:1]	0100: One-count mode
TAUJnMD0	0: Disables the start trigger during operation

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-34 TAUJnCMURm settings overflow interrupt output function (during TAUJnTTINm width measurement)

Bit name	Setting
TAUJnTIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

Because the channel output mode is not used for this function, clear TAUJnTOE.TAUJnTOEm to 0. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the overflow interrupt output function (during TAUJnTTINm width measurement). Therefore, these registers must be set to 0.

(4) Operating procedure for overflow interrupt output function

Table 17-35 Simultaneous rewrite settings for overflow interrupt output function (during TAUJnTTINm width measurement)

Bit name	Setting
TAUJnRDE.TAUJnRDEm	0: Disables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.TAUJnRDEm = 0), set these bits to 0

(during TAUJnTTINm width measurement)

Table 17-36 Operating procedure for overflow interrupt output function (during TAUJnTTINm width measurement)

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 17-33 "TAUJnCMORm settings for overflow interrupt output function (during TAUJnTTINm width measurement)" on page 1233* and *Table 17-34 "TAUJnCMURm settings overflow interrupt output function (during TAUJnTTINm width measurement)" on page 1233*. Set the value of the TAUJnCDRm register.	Channel operation is stopped.
Start operation	Set TAUJnTS.TAUJnTSm to 1. TAUJnTS.TAUJnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUJnTTINm start edge	TAUJnTE.TAUJnTEm is set to 1 and TAUJnCNTm waits for detection of the start edge. When a start edge is detected, TAUJnCNTm loads the TAUJnCDRm value (FFFF $FFFF_H$).
During operation	The TAUJnCNTm register can be read at any time. Detection of TAUJnTTINm edges	TAUJnCNTm counts down. When the counter reaches 0000 0000_H: • INTTAUJnIm is generated. When a reverse edge of TAUJnTTINm is detected during count operation: • TAUJnCNTm stops counting and waits for a trigger. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TAUJnTTm to 1. TAUJnTT.TAUJnTTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is cleared to 0 and the counter stops. TAUJnCNTm stops and retains its current value.

Restart

17.16.4 TAUJnTTINm input period count detection function

(1) Overview

Summary This function measures the cumulative width of a TAUJnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture & gate count mode, see *Table 17-37 "TAUJnCMORm settings for TAUJnTTINm input period count detection function" on page 1237*.
- TAUJnTTOUTm is not used for this function.

Description The counter is enabled by setting the channel trigger bit (TAUJnTS.TAUJnTSm) to 1. This in turn sets TAUJnTE.TAUJnTEm = 1, enabling count operation. The counter awaits a valid TAUJnTTINm input edge.

When a valid TAUJnTTINm input start edge is detected, the counter starts counting from 0000 0000_H.

When a valid TAUJnTTINm input stop edge is detected, the current TAUJnCNTm value is loaded to TAUJnCDRm and an interrupt (INTTAUJnIm) is generated. The counter stops and retains its value (TAUJnCDRm + 1) until the next valid TAUJnTTINm input start edge is detected.

When the next valid TAUJnTTINm input start edge is detected, the counter resumes counting, starting with its value upon stopping.

If the counter reaches FFFF $FFFF_H$, the TAUJnCSRm.TAUJnOVF bit is set to 1 and the counter restarts counting from 0000 0000_H. The value of TAUJnCSRm.TAUJnOVF is reset by the CPU by setting TAUJnCSCm.TAUJnCLOV = 1.

Note The input TAUJnTTINm is sampled at the frequency of the operation clock, specified by the TAUJnCMORm.TAUJnCKS[1:0] bits.

Conditions The valid start and stop edges are specified by the TAUJnCMURm.TAUJnTIS[1:0] bits:
- If TAUJnCMURm.TAUJnTIS[1:0] = 10_B, the TAUJnTTINm input low period is counted. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUJnCMURm.TAUJnTIS[1:0] = 11_B, the TAUJnTTINm input high period is counted. The start trigger is a rising edge and the stop trigger is a falling edge.

(2) Equations

Cumulative TAUJnTTINm input width =
count clock cycle × [((FFFF $FFFF_H$ + 1) × TAUJnCSRm.TAUJnOVF) + (TAUJnCDRm capture value + 1)]

(3) Block diagram and general timing diagram

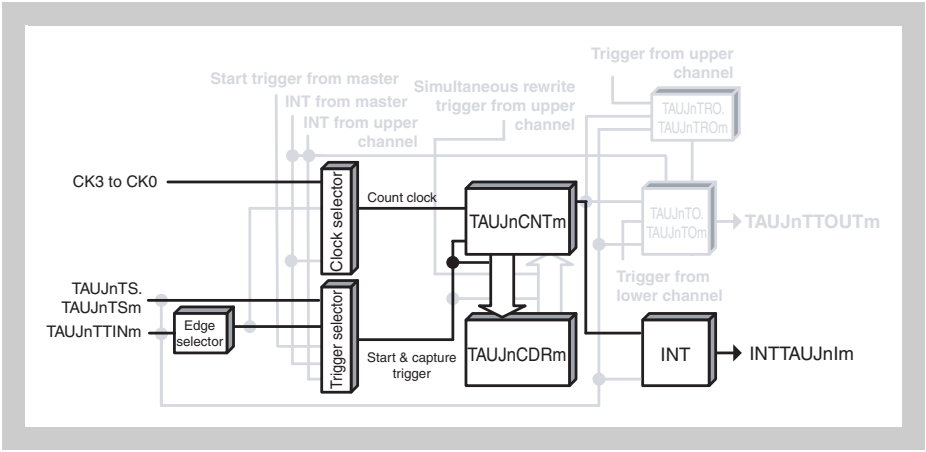


Figure 17-43 Block diagram for TAUJnTTINm input period count detection function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUJnCMURm.TAUJnTIS[1:0] = 11_B)

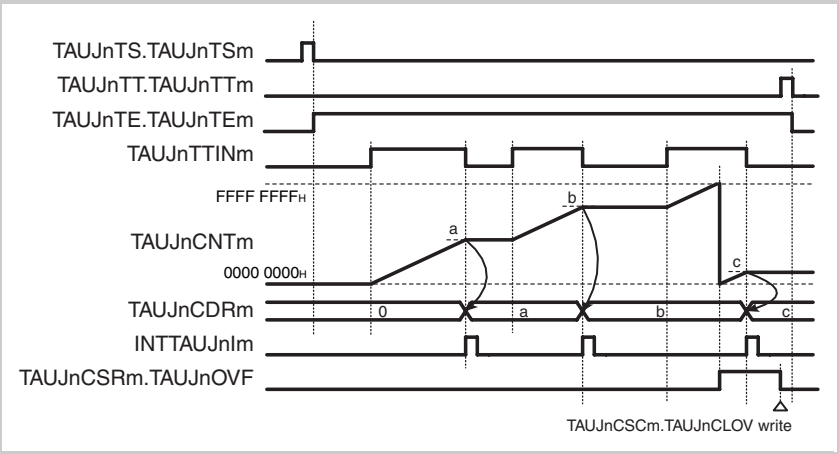


Figure 17-44 General timing diagram for TAUJnTTINm input period count detection function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-37 TAUJnCMORm settings for TAUJnTTINm input period count detection function

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output = CK0 01: Prescaler output = CK1 10: Prescaler output = CK2 11: Prescaler output = CK3
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	0: Not used, so set to 0
TAUJnSTS[2:0]	010: Valid edge of the TAUJnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
TAUJnCOS[1:0]	01: Overflow (TAUJnCSRm.TAUJnOVF) set upon counter overflow and cleared by a CPU instruction (by setting the TAUJnCSCm.TAUJnCLOV bit to 1).
TAUJnMD[4:1]	1101: Capture & gate count mode
TAUJnMD0	0: Disables the start trigger during operation

<R>

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-38 TAUJnCMURm settings for TAUJnTTINm input period count detection function

Bit name	Setting
TAUJnTIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

Because the channel output mode is not used for this function, clear TAUJnTOE.TAUJnTOEm to 0. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm input period count detection function. Therefore, these registers must be set to 0.

Table 17-39 Simultaneous rewrite settings for TAUJnTTINm input period count detection function

Bit name	Setting
TAUJnRDE.TAUJnRDEm	0: Disables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.TAUJnRDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm input period count detection function

Table 17-40 Operating procedure for TAUJnTTINm Input Period Count Detection Function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 17-37 "TAUJnCMORm settings for TAUJnTTINm input period count detection function" on page 1237* and *Table 17-38 "TAUJnCMURm settings for TAUJnTTINm input period count detection function" on page 1237*. Set the value of the TAUJnCDRm register.	Channel operation is stopped.
Start operation	Set TAUJnTS.TAUJnTSm to 1. TAUJnTS.TAUJnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUJnTTINm start edge	TAUJnTE.TAUJnTEm is set to 1 and TAUJnCNTm waits for detection of the TAUJnTTINm start edge. When a start edge is detected, TAUJnCNTm is cleared to 0000 0000_H and TAUJnCNTm starts counting up.
During operation	Detection of TAUJnTTINm edges The TAUJnCDRm, TAUJnCNTm, and TAUJnCSRm registers can be read at any time. The TAUJnCSCm.TAUJnCLOV bit can be set to 1.	When a TAUJnTTINm start edge (rising edge for high width measurement, falling edge for low width measurement) is detected, TAUJnCNTm starts counting up from the stop value. When TAUJnCNTm detects a stop edge (falling edge for high width measurement, rising edge for low width measurement), it transfers the value to TAUJnCDRm and INTTAUJnIm is generated. Counting stops at the "value transferred to TAUJnCDRm + 1" value and TAUJnCNTm waits for detection of the TAUJnTTINm start edge. If the TAUJnCNTm reaches FFFF $FFFF_H$, the counter overflows and TAUJnCSRm.TAUJnOVF is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TAUJnTTm to 1. TAUJnTT.TAUJnTTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is cleared to 0 and the counter stops. TAUJnCNTm stops and it and TAUJnCSRm.TAUJnOVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

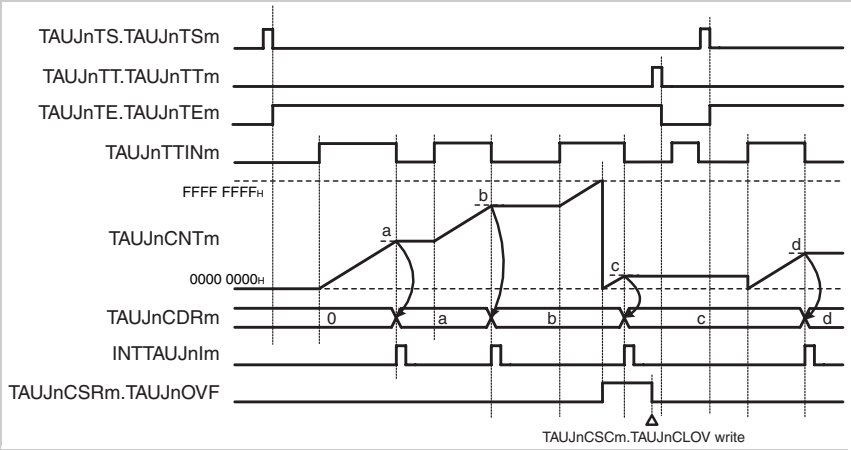


Figure 17-45 Operation stop and restart, TAUJnCMURm.TAUJnTIS[1:0] = 11_B

- The counter can be stopped by setting TAUJnTT.TAUJnTTm to 1, which in turn sets TAUJnTE.TAUJnTEm to 0.
- TAUJnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUJnTTINm input edges are ignored.
- The counter can be restarted by setting TAUJnTS.TAUJnTSm to 1. TAUJnCNTm restarts counting from 0000 0000_H.

17.16.5 Overflow interrupt output function (during TAUJnTTINm input period count detection)

(1) Overview

Summary This function measures the cumulative width of a TAUJnTTINm input signal. If the cumulative TAUJnTTINm input width is longer than FFFF $FFFF_H$, an interrupt can be generated to output an overflow interrupt.

Prerequisites

- The operation mode must be set to gate count mode, see *Table 17-41 "TAUJnCMORm settings for overflow interrupt output function (during TAUJnTTINm input period count detection)" on page 1243*.
- TAUJnTTOUTm is not used for this function.
- The value of TAUJnCDRm must be set to FFFF $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUJnTS.TAUJnTSm) to 1. This in turn sets TAUJnTE.TAUJnTEm = 1, enabling count operation.

The counter starts when a valid TAUJnTTINm input start edge is detected. FFFF $FFFF_H$ is loaded to TAUJnCNTm and the counter starts counting down.

When a valid stop edge is detected, the counter stops and retains the current value. The counter awaits the next TAUJnTTINm input start edge and then continues counting down from the current value.

When the counter reaches 0000 0000_H an interrupt is generated. FFFF $FFFF_H$ is loaded to TAUJnCNTm and the counter continues counting down until a TAUJnTTINm input stop edge is detected.

Conditions The valid start and stop edges are specified by the TAUJnCMURm.TAUJnTIS[1:0] bits:

- If TAUJnCMURm.TAUJnTIS[1:0] = 10_B, the TAUJnTTINm input low period is counted. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUJnCMURm.TAUJnTIS[1:0] = 11_B, the TAUJnTTINm input high period is counted. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

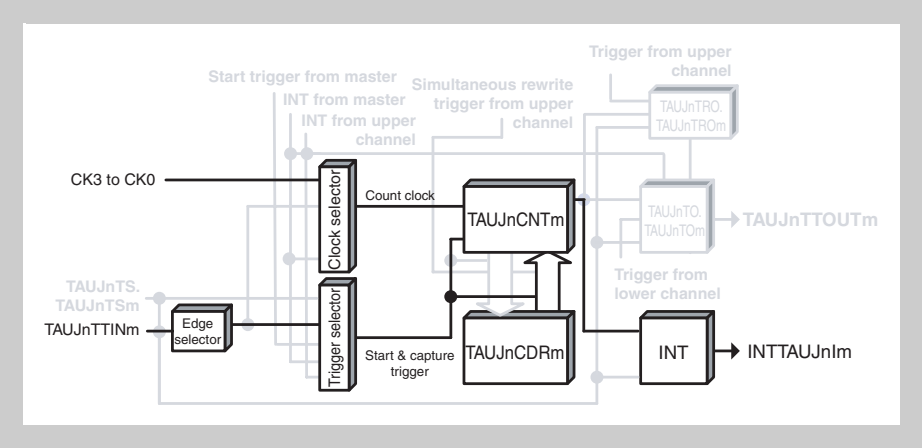


Figure 17-46 Block diagram for overflow interrupt output function (during TAUJnTTINm input period count detection)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUJnCMURm.TAUJnTIS[1:0] = 11_B)

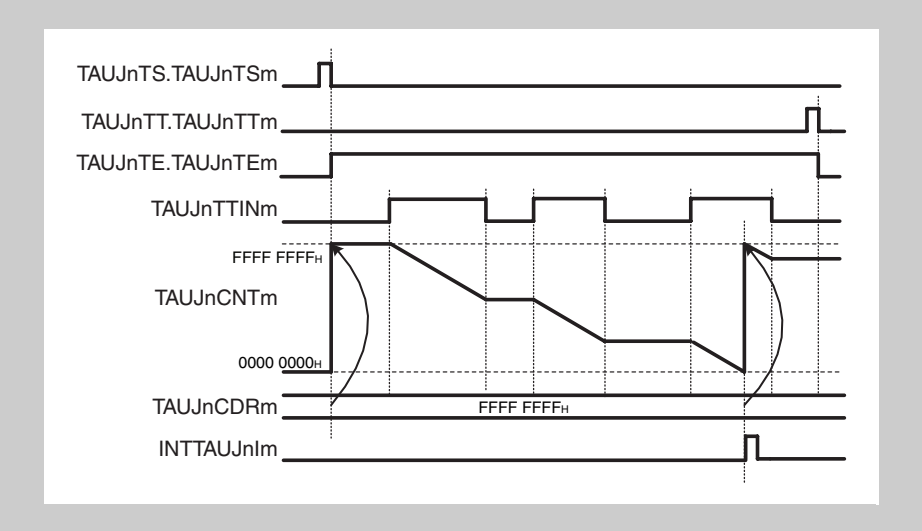


Figure 17-47 General timing diagram for overflow interrupt output function (during TAUJnTTINm input period count detection)

(3) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-41 TAUJnCMORm settings for overflow interrupt output function (during TAUJnTTINm input period count detection)

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output = CK0 01: Prescaler output = CK1 10: Prescaler output = CK2 11: Prescaler output = CK3
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	0: Not used, so set to 0
TAUJnSTS[2:0]	010: Valid edge of the TAUJnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
TAUJnCOS[1:0]	00: Not used, so set to 00
TAUJnMD[4:1]	1100: Gate count mode
TAUJnMD0	0: INTTAUJnIm not generated at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-42 TAUJnCMURm settings for overflow interrupt output function (during TAUJnTTINm input period count detection)

Bit name	Setting
TAUJnTIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

Because the channel output mode is not used for this function, clear TAUJnTOE.TAUJnTOEm to 0. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the overflow interrupt output function (during TAUJnTTINm input period count detection). Therefore, these registers must be set to 0.

Table 17-43 Simultaneous rewrite settings for overflow interrupt output function (during TAUJnTTINm input period count detection)

Bit name	Setting
TAUJnRDE.TAUJnRDEm	0: Disables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.TAUJnRDEm = 0), set these bits to 0

(4) Operating procedure for overflow interrupt output function (during TAUJnTTINm input period count detection)

Table 17-44 Operating procedure for overflow interrupt output function (during TAUJnTTINm input period count detection)

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 17-41 "TAUJnCMORm settings for overflow interrupt output function (during TAUJnTTINm input period count detection)" on page 1243* and *Table 17-42 "TAUJnCMURm settings for overflow interrupt output function (during TAUJnTTINm input period count detection)" on page 1243*. Set the value of the TAUJnCDRm register.	Channel operation is stopped.
Start operation	Set TAUJnTS.TAUJnTSm to 1. TAUJnTS.TAUJnTSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUJnTTINm start edge	TAUJnTE.TAUJnTEm is set to 1 and TAUJnCNTm waits for detection of the start edge. When a start edge is detected, TAUJnCNTm loads the TAUJnCDRm value (FFFF FFFF$_H$).
During operation	The TAUJnCNTm register can be read at all times.	TAUJnCNTm counts down. When the counter reaches 0000 0000$_H$: • INTTAUJnIm is generated. • TAUJnCNTm loads the TAUJnCDRm value (FFFF FFFF$_H$) and continues counting down. When a reverse edge of TAUJnTTINm is detected during count operation: • TAUJnCNTm counts down from the stop value. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TAUJnTTm to 1. TAUJnTT.TAUJnTTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is cleared to 0 and the counter stops. TAUJnCNTm stops and retains its current value.

Restart

17.17 Other Independent Channel Function

This section describes a function that measures the duration between the function start and a TAUJnTTINm input signal.

17.17.1 TAUJnTTINm input position detection function

(1) Overview

Summary This function measures the duration between the function start and a TAUJnTTINm input signal.

Prerequisites

- The operation mode must be set to count capture mode, see *Table 17-45 "TAUJnCMORm settings for TAUJnTTINm input position detection function" on page 1249*.
- TAUJnTTOUTm is not used for this function.

Description The counter operation is enabled by setting the channel trigger bit (TAUJnTS.TAUJnTSm) to 1. This in turn sets TAUJnTE.TAUJnTEm = 1, enabling count operation. The counter starts counting from 0000 0000_H. When a valid TAUJnTTINm input edge is detected, the current TAUJnCNTm value is loaded to TAUJnCDRm and an interrupt (INTTAUJnIm) is generated. The counter continues counting.

When the counter reaches FFFF $FFFF_H$, the TAUJnCSRm.TAUJnOVF bit is set to 1 and the counter restarts counting from 0000 0000_H.

TAUJnCSRm.TAUJnOVF is cleared by writing 1 to TAUJnCSCm.TAUJnCLOV.

Conditions If the TAUJnCMORm.TAUJnMD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details, see *17.11 "TAUJnTTOUTm Output and INTTAUJnIm Generation When Counter Starts or Restarts (by TAUJnMD0 Bit)" on page 1195*.

(2) Equations

Function duration at a TAUJnTTINm input pulse =

count clock cycle × [(FFFF $FFFF_H$+1 × TAUJnCSRm.TAUJnOVF) + (TAUJnCDRm capture value + 1)]

(3) Block diagram and general timing diagram

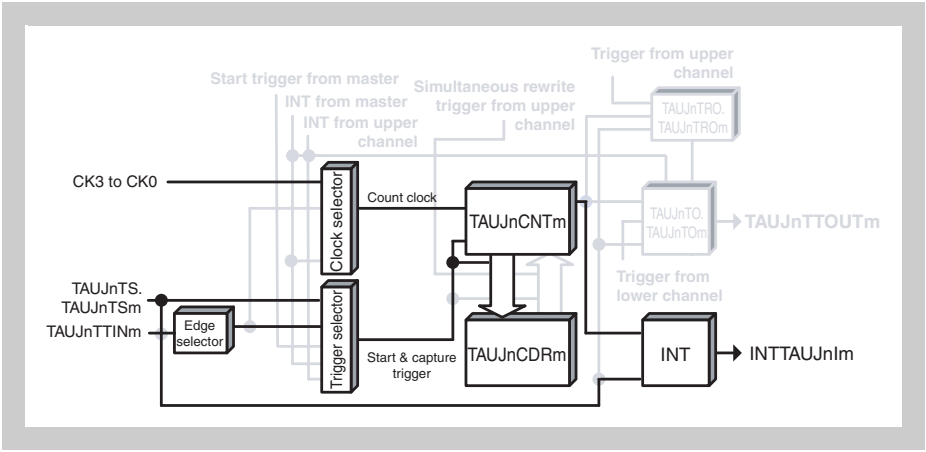


Figure 17-48 Block diagram for TAUJnTTINm input position detection function

The following settings apply to the general timing diagram:

- INTTAUJnIm not generated at operation start (TAUJnCMORm.TAUJnMD0 = 0)
- Falling edge detection (TAUJnCMURm.TAUJnTIS[1:0] = 00_B)

<R>

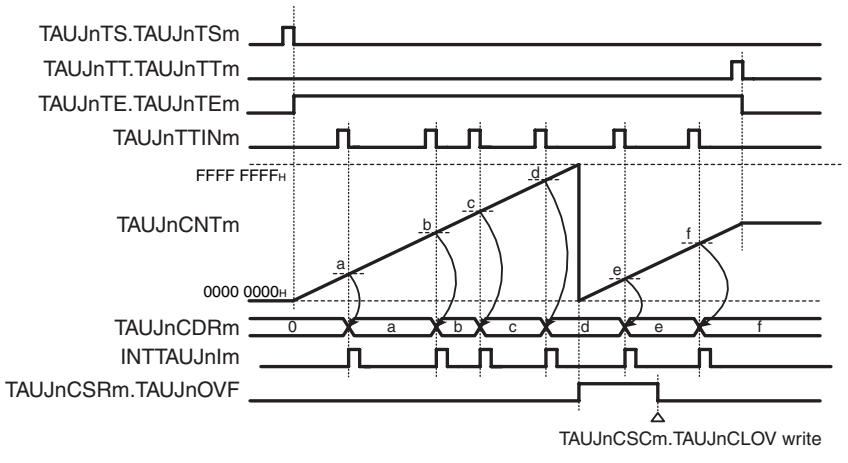


Figure 17-49 General timing diagram for TAUJnTTINm input position detection function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-45 TAUJnCMORm settings for TAUJnTTINm input position detection function

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output = CK0 01: Prescaler output = CK1 10: Prescaler output = CK2 11: Prescaler output = CK3
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	0: Not used, so set to 0
TAUJnSTS[2:0]	001: Valid TAUJnTTINm input edge signal is used as the external capture trigger
TAUJnCOS[1:0]	01: Overflow (TAUJnCSRm.TAUJnOVF) set upon counter overflow and cleared by a CPU instruction (by setting the TAUJnCSCm.TAUJnCLOV bit to 1).
TAUJnMD[4:1]	1011: Count capture mode
TAUJnMD0	0: INTTAUJnIm not generated at operation start 1: Generates INTTAUJnIm at operation start

<R>

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-46 TAUJnCMURm settings for TAUJnTTINm input position detection function

Bit name	Setting
TAUJnTIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in independent channel output mode controlled by software.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm input position detection function. Therefore, these registers must be set to 0.

Table 17-47 Simultaneous rewrite settings for TAUJnTTINm input position detection function

Bit name	Setting
TAUJnRDE.TAUJnRDEm	0: Disables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.TAUJnRDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm input position detection function

Table 17-48 Operating procedure for TAUJnTTINm input position detection function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 17-45 "TAUJnCMORm settings for TAUJnTTINm input position detection function" on page 1249* and *Table 17-46 "TAUJnCMURm settings for TAUJnTTINm input position detection function" on page 1249*. Set the value of the TAUJnCDRm register.	Channel operation is stopped.
Start operation	Set TAUJnTS.TAUJnTSm to 1. TAUJnTS.TAUJnTSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is set to 1 and the counter starts. INTTAUJnIm is generated when TAUJnCMORm.TAUJnMD0 is set to 1.
During operation	The TAUJnCMURm.TAUJnTIS[1:0] bits can be changed at any time. The TAUJnCDRm and TAUJnCSRm registers can be read at any time. The TAUJnCSC.CLOV bit can be set to 1.	TAUJnCNTm starts counting up from $0000\ 0000_H$. When a TAUJnTTINm valid edge is detected: • TAUJnCNTm transfers (captures) its value to TAUJnCDRm. • INTTAUJnIm is output. • The counter value is not cleared to $0000\ 0000_H$ and TAUJnCNTm continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TAUJnTTm to 1. TAUJnTT.TAUJnTTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is cleared to 0 and the counter stops. TAUJnCNTm stops and both it and TAUJnCSRm.TAUJnOVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

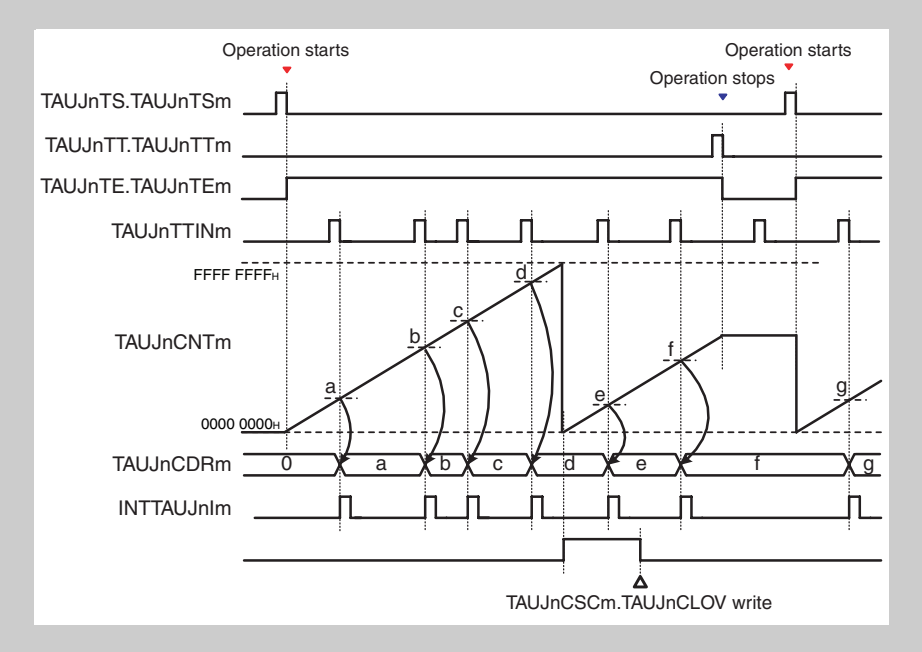


Figure 17-50 Operation stop and restart, TAUJnCMORm.TAUJnMD0 = 0, TAUJnCMURm.TAUJnTIS[1:0] = 00

- The counter can be stopped by setting TAUJnTT.TAUJnTTm to 1, which in turn sets TAUJnTE.TAUJnTEm to 0.
- TAUJnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUJnTTINm input edges are ignored.
- The counter can be restarted by setting TAUJnTS.TAUJnTSm to 1. TAUJnCNTm restarts counting from 0000 0000_H.

17.18 Synchronous PWM Signal Functions Triggered at Regular Intervals

This section describes a function that generates PWM signals at regular intervals. For a general overview of synchronous channel operation, see *17.4 “Functional Description” on page 1177*.

17.18.1 PWM output function

(1) Overview

Summary This function generates multiple PWM outputs by using a master and multiple slave channels. It enables the pulse cycle (frequency) and the duty of the TAUJnTTOUTm to be set. The pulse cycle is set in the master channel. The duty is set in the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to interval timer mode, see *Table 17-49 "TAUJnCMORm settings for the master channel of the PWM output function" on page 1257*.
- The operation mode of the slave channel(s) must be set to one-count mode, see *Table 17-52 "TAUJnCMORm settings for the slave channel of the PWM output function" on page 1258*.
- TAUJnTTOUTm is not used for the master channel of this function.
- The channel output mode of the slave channel(s) must be set to synchronous channel output mode 1 (*17.9 "Channel output modes" on page 1188*).

Description The counters are started by setting the channel trigger bits (TAUJnTS.TAUJnTSm) to 1. This in turn sets TAUJnTE.TAUJnTEm = 1, enabling count operation. The current value of TAUJnCDRm is loaded to TAUJnCNTm and the counters start counting down from these values. INTTAUJnIm is generated on the master channel, and PWM output is achieved by setting and resetting TAUJnTTOUTm (slave).

- Master channel:

  When the counter of the master channel reaches 0000 0000_H, pulse cycle time has elapsed and INTTAUJnIm is generated. TAUJnCNTm loads the TAUJnCDRm value, and then counts down.

- Slave channel(s)

  The INTTAUJnIm of the master channel triggers the counter operation of the slave channel(s). The current value of TAUJnCDRm (slave) is loaded to TAUJnCNTm (slave) and the counter starts counting down from this value. The TAUJnTTOUTm signal becomes active.

  When the counter reaches 0000 0000_H, i.e., duty time has elapsed, INTTAUJnIm is generated and the TAUJnTTOUTm signal becomes inactive. The counter returns to FFFF $FFFF_H$ and awaits the next INTTAUJnIm of the master channel, and thus the start of the next pulse cycle.

The counter can be stopped by setting TAUJnTT.TAUJnTTm to 1 for the master and slave channel(s), which in turn sets TAUJnTE.TAUJnTEm to 0. TAUJnCNTm and TAUJnTTOUTm of master and slave channel(s) stop but retain their values. The counters can be restarted by setting TAUJnTS.TAUJnTSm to 1.

Conditions Simultaneous rewrite can be used with this function. See *17.8 "Simultaneous Rewrite" on page 1183*.

(2) Equations

Pulse cycle = (TAUJnCDRm (master) + 1) x count clock cycle

Duty cycle [%] = (TAUJnCDRm (slave) / (TAUJnCDRm (master) + 1)) x 100

– Duty cycle = 0 %

TAUJnCDRm (slave) = 0000 0000_H

– Duty cycle = 100 %

TAUJnCDRm (slave) ≥ TAUJnCDRm (master) + 1

(3) Block diagram and general timing diagram

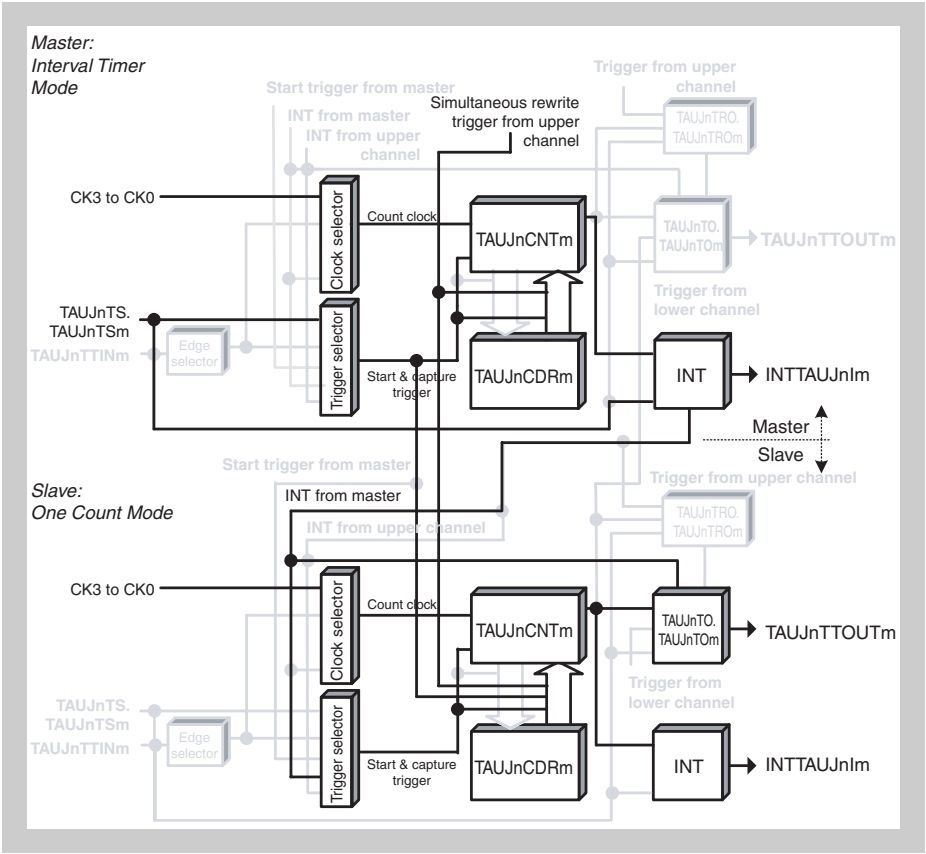


Figure 17-51 Block diagram for PWM output function

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUJnTOL.TAUJnTOLm = 0)

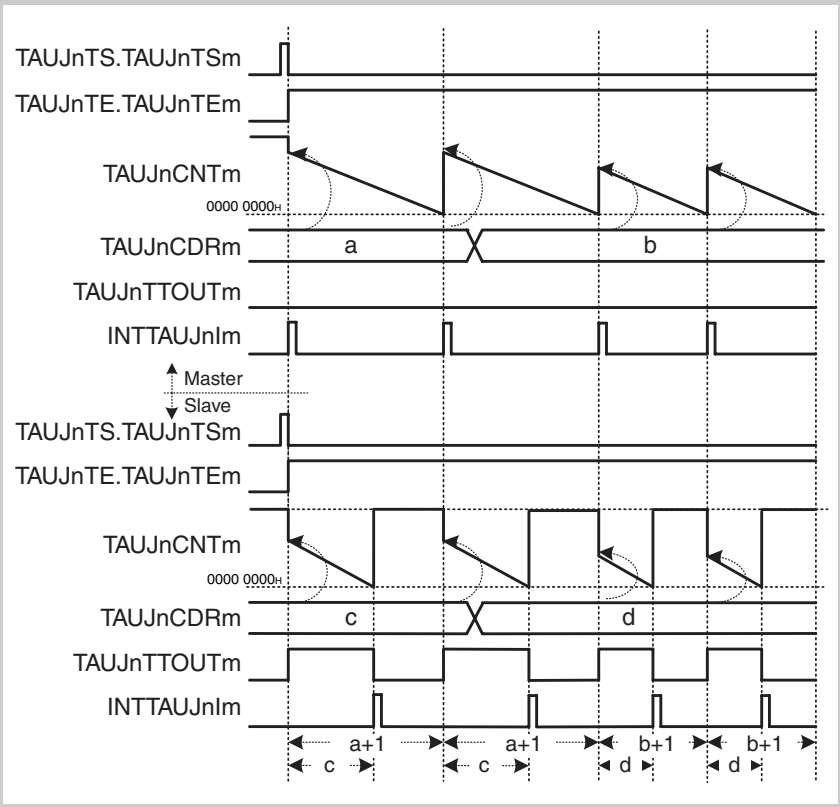


Figure 17-52 General timing diagram for PWM output function

Note The interval between the slave channel starting counting and an interrupt being generated is the value of corresponding TAUJnCDRm, whereas for the master channel the interval is the corresponding TAUJnCDRm + 1.

(4) Register settings for the master channel

(a) TAUJnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-49 TAUJnCMORm settings for the master channel of the PWM output function

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUJnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	1: Channel is master channel
TAUJnSTS[2:0]	000: Counter triggered by software trigger
TAUJnCOS[1:0]	00: Not used, so set to 00
TAUJnMD[4:1]	0000: Interval timer mode
TAUJnMD0	1: Generates INTTAUJnIm at operation start

(b) TAUJnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-50 TAUJnCMURm settings for the master channel of the PWM output function

Bit name	Setting
TAUJnTIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in independent channel output mode controlled by software.

<R> **(d) Simultaneous rewrite for the master channel**

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 17-51 Simultaneous rewrite settings for the master channel of the PWM output function

Bit name	Setting
TAUJnRDE.TAUJnRDEm	1: Enables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting

(5) Register settings for the slave channel(s)

(a) TAUJnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:1]				TAUJn MD0

Table 17-52 TAUJnCMORm settings for the slave channel of the PWM output function

Bit name	Setting
TAUJnCKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the TAUJnCKS[1:0] bit of the master and slave channel(s) must be identical.
TAUJnCCS[1:0]	00: Operation clock is used as the count clock
TAUJnMAS	0: Channel is a slave channel
TAUJnSTS[2:0]	100: INTTAUJnIm of the master channel is the start trigger
TAUJnCOS[1:0]	00: Not used, so set to 00
TAUJnMD[4:1]	0100: One-count mode
TAUJnMD0	1: Enables the start trigger during operation

(b) TAUJnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TAUJnTIS[1:0]	

Table 17-53 TAUJnCMURm settings for the slave channel of the PWM output function

Bit name	Setting
TAUJnTIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel(s)

Table 17-54 Control bit settings for independent channel output mode 1

Bit name	Setting
TAUJnTOE.TAUJnTOEm	1: Enables independent channel output mode
TAUJnTO.TAUJnTOm	0: Low level 1: High level
TAUJnTOM.TAUJnTOMm	1: Synchronous channel operation
TAUJnTOC.TAUJnTOCm	0: Operation mode 1
TAUJnTOL.TAUJnTOLm	0: Positive logic 1: Inverted logic

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 17-55 Simultaneous rewrite settings for the slave channel of the PWM output function

Bit name	Setting
TAUJnRDE.TAUJnRDEm	1: Enables simultaneous rewrite
TAUJnRDM.TAUJnRDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting

(6) Operating procedure for PWM output function

Table 17-56 Operating procedure for PWM output function

	Operation	Status of TAUJn
Initial channel setting	Master channel: set the TAUJnCMORm and TAUJnCMURm registers and the channel output mode as described in *(4) "Register settings for the master channel" on page 1257*. Slave channel: set the TAUJnCMORm and TAUJnCMURm registers and the channel output mode as described in *(5) "Register settings for the slave channel(s)" on page 1258*. Set the values of the TAUJnCDRm registers of all channels.	Channel operation is stopped.
Start operation	Set TAUJnTS.TAUJnTSm of the master and slave channels to 1 simultaneously. TAUJnTS.TAUJnTSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUJnIm is generated on the master channel and TAUJnTTOUTm (slave) is set.
During operation	TAUJnCDRm can be changed at any time. TAUJnCNTm and TAUJnRSF.TAUJnRSFm can be read at any time. TAUJnRDT.TAUJnRDTm can be changed during operation.	TAUJnCNTm of the master channel loads TAUJnCDRm and counts down. When the counter reaches 0000 0000_H: • INTTAUJnIm (master) is generated. • TAUJnCNTm (master) loads the TAUJnCDRm value, and then continues count operation. • TAUJnCNTm (slave) loads the TAUJnCDRm value, and then counts down. • TAUJnTTOUTm (slave) becomes active. When TAUJnCNTm (slave) reaches 0000 0000_H: • INTTAUJnIm (slave) is generated. • TAUJnTTOUTm (slave) becomes inactive.
Stop operation	Set TAUJnTT.TAUJnTTm of the master and slave channels to 1 simultaneously. TAUJnTT.TAUJnTTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TAUJnTEm is cleared to 0 and the counter stops. TAUJnCNTm and TAUJnTTOUTm stop and retain their current values.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0 %

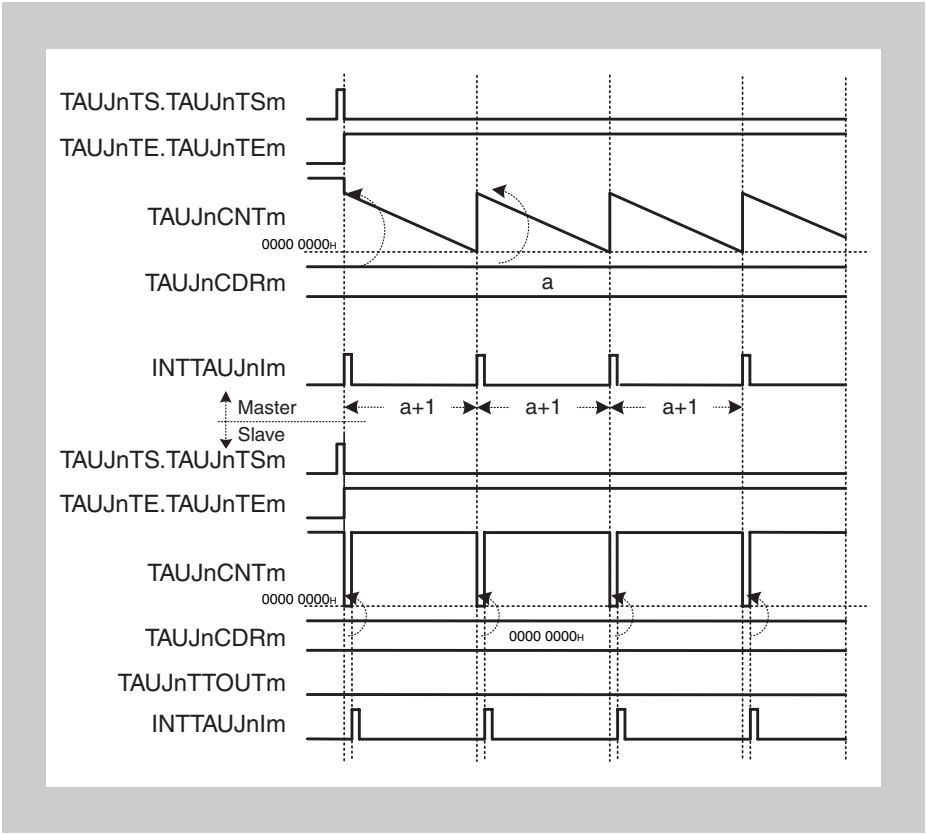


Figure 17-53 TAUJnCDRm (slave) = 0000 0000_H, positive logic (TAUJnTOL.TAUJnTOLm (slave) = 0)

- Every time the master channel generates an interrupt (INTTAUJnIm), 0000 0000_H is loaded to TAUJnCNTm (slave). Therefore, TAUJnCNTm (slave) cannot start counting and TAUJnTTOUTm remains at not active state.
- TAUJnCNTm (slave) loads the TAUJnCDRm value and an interrupt is generated.

(b) Duty cycle = 100 %

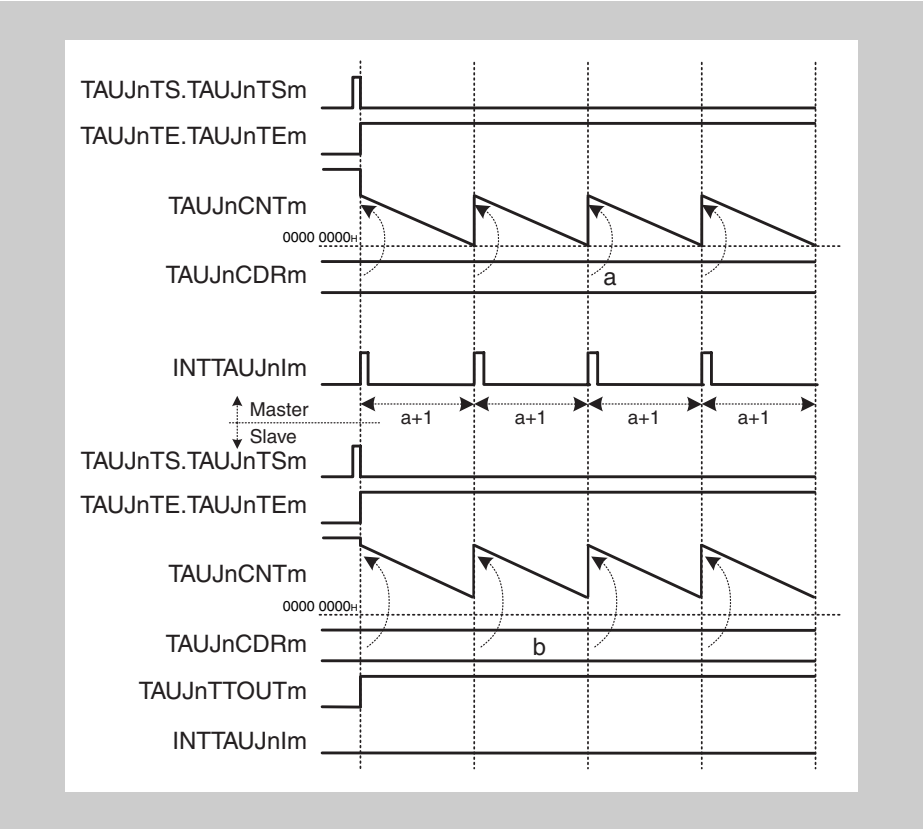


Figure 17-54 TAUJnCDRm (slave) ≥ TAUJnCDRm (master) + 1, positive logic (TAUJnTOL.TAUJnTOLm (slave) = 0)

- If the TAUJnCDRm (slave) value is higher than the TAUJnCDRm (master) value, the counter of the slave channel cannot reach 0000 0000_H and interrupts are therefore not generated. The TAUJnTTOUTm remains at active state.

(c) Stop and restart operation

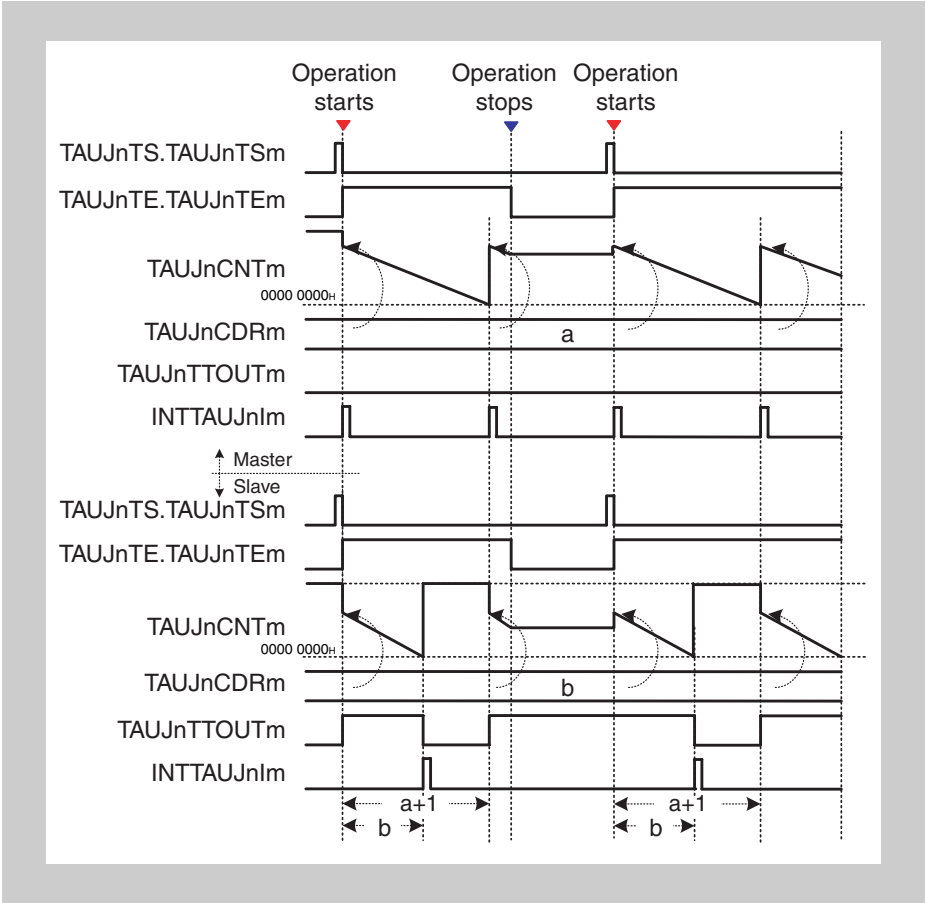


Figure 17-55 Stop and restart operation
Positive logic (TAUJnTOL.TAUJnTOLm (slave) = 0)

- The counter can be stopped by setting TAUJnTT.TAUJnTTm of the master and slave channel(s) to 1, which in turn sets TAUJnTE.TAUJnTEm to 0.
- TAUJnCNTm and TAUJnTTOUTm of all channels stop and the current values are retained. No interrupts are generated.
- The counter can be restarted by setting TAUJnTS.TAUJnTSm of master and slave channel(s) to 1.
  TAUJnCNTm loads the TAUJnCDRm value of master and slave channels, and then starts counting down from this value.

17.19 Registers

This section contains a description of all the registers of the 32-bit TAUJ.

17.19.1 TAUJn register overview

The TAUJn is controlled and operated by the registers in the following table. Where there is one register per channel, this is indicated by an “m”, where m runs from 0 to 3.

Table 17-57 TAUJn register overview

Register name	Symbol	Address
TAUJn prescaler registers		
TAUJn prescaler clock select register	TAUJnTPS	<TAUJn_base> + 90_H
TAUJn prescaler baud rate setting register	TAUJnBRS	<TAUJn_base> + 94_H
TAUJn control registers		
TAUJn channel data register m	TAUJnCDRm	<TAUJn_base> + m x 4_H
TAUJn channel counter register m	TAUJnCNTm	<TAUJn_base> + 10_H + m x 4_H
TAUJn channel mode OS register m	TAUJnCMORm	<TAUJn_base> + 80_H + m x 4_H
TAUJn channel mode user register m	TAUJnCMURm	<TAUJn_base> + 20_H + m x 4_H
TAUJn channel status register m	TAUJnCSRm	<TAUJn_base> + 30_H + m x 4_H
TAUJn channel status clear trigger register m	TAUJnCSCm	<TAUJn_base> + 40_H + m x 4_H
TAUJn channel start trigger register	TAUJnTS	<TAUJn_base> + 54_H
TAUJn channel enable status register	TAUJnTE	<TAUJn_base> + 50_H
TAUJn channel stop trigger register	TAUJnTT	<TAUJn_base> + 58_H
TAUJn output registers		
TAUJn channel output enable register	TAUJnTOE	<TAUJn_base> + 60_H
TAUJn channel output register	TAUJnTO	<TAUJn_base> + $5C_H$
TAUJn channel output mode register	TAUJnTOM	<TAUJn_base> + 98_H
TAUJn channel output configuration register	TAUJnTOC	<TAUJn_base> + $9C_H$
TAUJn channel output active level register	TAUJnTOL	<TAUJn_base> + 64_H
TAUJn reload data registers		
TAUJn channel reload data enable register	TAUJnRDE	<TAUJn_base> + $A0_H$
TAUJn channel reload data mode register	TAUJnRDM	<TAUJn_base> + $A4_H$
TAUJn channel reload data trigger register	TAUJnRDT	<TAUJn_base> + 68_H
TAUJn channel reload status register	TAUJnRSF	<TAUJn_base> + $6C_H$

Note The <TAUJn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

17.19.2 TAUJn prescaler register details

(1) TAUJnTPS - TAUJn prescaler clock select register

This register specifies the PCLK prescalers for clocks CK0, CK1, CK2, and CK3_PRE for all channels. CK3 is generated by dividing CK3_PRE by the factor specified in TAUJnBRS.

Access This register can be read/written in 16-bit units.

Address <TAUJn_base> + 90_H

Initial Value $FFFF_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJnPRS3[3:0]				TAUJnPRS2[3:0]				TAUJnPRS1[3:0]				TAUJnPRS0[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-58 TAUJnTPS register contents (1/3)

Bit position	Bit name	Function
15 to 12	TAUJnPRS3[3:0]	Specifies the CK3_PRE clock. Clock CK3_PRE is the input clock of the BRG unit. The BRG unit supplies the CK3 operation clock for all channels.

TAUJnPRS3[3:0]	CK3_PRE clock
0000_B	PCLK/2^0
0001_B	PCLK/2^1
0010_B	PCLK/2^2
0011_B	PCLK/2^3
0100_B	PCLK/2^4
0101_B	PCLK/2^5
0110_B	PCLK/2^6
0111_B	PCLK/2^7
1000_B	PCLK/2^8
1001_B	PCLK/2^9
1010_B	PCLK/2^{10}
1011_B	PCLK/2^{11}
1100_B	PCLK/2^{12}
1101_B	PCLK/2^{13}
1110_B	PCLK/2^{14}
1111_B	PCLK/2^{15}

These bits can only be rewritten when all counters using CK3 are stopped (TAUJnTE.TAUJnTEm = 0).

Table 17-58 TAUJnTPS register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>11 to 8</td><td>TAUJnPRS2[3:0]</td><td>Specifies the CK2 clock.
<table>
<tr><th>PRS2[3:0]</th><th>CK2 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>0100_B</td><td>PCLK/2^4</td></tr>
<tr><td>0101_B</td><td>PCLK/2^5</td></tr>
<tr><td>0110_B</td><td>PCLK/2^6</td></tr>
<tr><td>0111_B</td><td>PCLK/2^7</td></tr>
<tr><td>1000_B</td><td>PCLK/2^8</td></tr>
<tr><td>1001_B</td><td>PCLK/2^9</td></tr>
<tr><td>1010_B</td><td>PCLK/2^{10}</td></tr>
<tr><td>1011_B</td><td>PCLK/2^{11}</td></tr>
<tr><td>1100_B</td><td>PCLK/2^{12}</td></tr>
<tr><td>1101_B</td><td>PCLK/2^{13}</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK2 are stopped (TAUJnTE.TAUJnTEm = 0).</td></tr>
<tr><td>7 to 4</td><td>TAUJnPRS1[3:0]</td><td>Specifies the CK1 clock.
<table>
<tr><th>PRS1[3:0]</th><th>CK1 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>0100_B</td><td>PCLK/2^4</td></tr>
<tr><td>0101_B</td><td>PCLK/2^5</td></tr>
<tr><td>0110_B</td><td>PCLK/2^6</td></tr>
<tr><td>0111_B</td><td>PCLK/2^7</td></tr>
<tr><td>1000_B</td><td>PCLK/2^8</td></tr>
<tr><td>1001_B</td><td>PCLK/2^9</td></tr>
<tr><td>1010_B</td><td>PCLK/2^{10}</td></tr>
<tr><td>1011_B</td><td>PCLK/2^{11}</td></tr>
<tr><td>1100_B</td><td>PCLK/2^{12}</td></tr>
<tr><td>1101_B</td><td>PCLK/2^{13}</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK1 are stopped (TAUJnTE.TAUJnTEm = 0).</td></tr>
</table>

Table 17-58 TAUJnTPS register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>11 to 8</td><td>TAUJnPRS2[3:0]</td><td>Specifies the CK2 clock.
<table>
<tr><th>PRS2[3:0]</th><th>CK2 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>0100_B</td><td>PCLK/2^4</td></tr>
<tr><td>0101_B</td><td>PCLK/2^5</td></tr>
<tr><td>0110_B</td><td>PCLK/2^6</td></tr>
<tr><td>0111_B</td><td>PCLK/2^7</td></tr>
<tr><td>1000_B</td><td>PCLK/2^8</td></tr>
<tr><td>1001_B</td><td>PCLK/2^9</td></tr>
<tr><td>1010_B</td><td>PCLK/2^{10}</td></tr>
<tr><td>1011_B</td><td>PCLK/2^{11}</td></tr>
<tr><td>1100_B</td><td>PCLK/2^{12}</td></tr>
<tr><td>1101_B</td><td>PCLK/2^{13}</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK2 are stopped (TAUJnTE.TAUJnTEm = 0).</td></tr>
<tr><td>7 to 4</td><td>TAUJnPRS1[3:0]</td><td>Specifies the CK1 clock.
<table>
<tr><th>PRS1[3:0]</th><th>CK1 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>0100_B</td><td>PCLK/2^4</td></tr>
<tr><td>0101_B</td><td>PCLK/2^5</td></tr>
<tr><td>0110_B</td><td>PCLK/2^6</td></tr>
<tr><td>0111_B</td><td>PCLK/2^7</td></tr>
<tr><td>1000_B</td><td>PCLK/2^8</td></tr>
<tr><td>1001_B</td><td>PCLK/2^9</td></tr>
<tr><td>1010_B</td><td>PCLK/2^{10}</td></tr>
<tr><td>1011_B</td><td>PCLK/2^{11}</td></tr>
<tr><td>1100_B</td><td>PCLK/2^{12}</td></tr>
<tr><td>1101_B</td><td>PCLK/2^{13}</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK1 are stopped (TAUJnTE.TAUJnTEm = 0).</td></tr>
</table>

Table 17-58 TAUJnTPS register contents (3/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>3 to 0</td><td>TAUJnPRS0[3:0]</td><td>Specifies the CK0 clock.
<table>
<tr><th>PRS0[3:0]</th><th>CK0 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>0100_B</td><td>PCLK/2^4</td></tr>
<tr><td>0101_B</td><td>PCLK/2^5</td></tr>
<tr><td>0110_B</td><td>PCLK/2^6</td></tr>
<tr><td>0111_B</td><td>PCLK/2^7</td></tr>
<tr><td>1000_B</td><td>PCLK/2^8</td></tr>
<tr><td>1001_B</td><td>PCLK/2^9</td></tr>
<tr><td>1010_B</td><td>PCLK/2^{10}</td></tr>
<tr><td>1011_B</td><td>PCLK/2^{11}</td></tr>
<tr><td>1100_B</td><td>PCLK/2^{12}</td></tr>
<tr><td>1101_B</td><td>PCLK/2^{13}</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK0 are stopped (TAUJnTE.TAUJnTEm = 0).</td></tr>
</table>

Note The TAUJn clock input PCLK is specified in the first section of this chapter under the keyword “Clock supply”.

(2) TAUJnBRS - TAUJn prescaler baud rate setting register

This register specifies the division factor of prescaler clock CK3.

CK3 is generated by dividing CK3_PRE by the factor specified in this register plus one. The PCLK prescaler for CK3_PRE is specified in TAUJnTPS.TAUJnPRS3[3:0].

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + 94_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
TAUJBRS[07:00]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-59 TAUJnBRS register contents

Bit position	Bit name	Function
7 to 0	TAUJnBRS[07:00]	Specifies the CK3_PRE clock division factor for generating CK3.

TAUJnBRS[07:00]	CK3 clock
$0000\ 0000_B$	CK3_PRE / 1
$0000\ 0001_B$	CK3_PRE / 2
$0000\ 0010_B$	CK3_PRE / 3
$0000\ 0011_B$	CK3_PRE / 4
...	...
$1111\ 1110_B$	CK3_PRE / 255
$1111\ 1111_B$	CK3_PRE / 256

17.19.3 TAUJn control register details

(1) TAUJnCDRm - TAUJn channel data register

This register functions either as a compare register or as a capture register, depending on the operation mode specified in TAUJnCMORm.TAUJnMD[4:0].

Access This register can be read/written in 32-bit or 16-bit units.

- In capture mode, only reading is possible. Write operation is ignored.
- In compare mode, reading and writing is possible.

Address <TAUJn_base> + 0_H + m x 4_H

Initial Value 0000 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TAUJnCDR[31:16]															
R/W															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJnCDR[15:0]															
R/W															

Table 17-60 TAUJnCDRm register contents

Bit position	Bit name	Function
31 to 0	TAUJnCDR[31:0]	Data register for the capture/compare value

(2) TAUJnCNTm - TAUJn channel counter register

This register is the channel m counter register.

Access This register can be read in 32-bit or 16-bit units.

Address <TAUJn_base> + 10_H + m x 4_H

Initial Value 0000 0000_H or FFFF $FFFF_H$. The initial value depends on the operation mode, see *Table 17-62 "TAUJnCNTm read values after the counter is re-enabled" on page 1272*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TAUJnCNT[31:16]															
R															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJnCNT[15:0]															
R															

Table 17-61 TAUJnCNTm register contents

Bit position	Bit name	Function
31 to 0	TAUJnCNT[31:0]	32-bit counter value

The read value depends on the counter, the operation mode change, and the values of the TAUJnTS.TAUJnTSm and TAUJnTT.TAUJnTTm bits.

The *initial* counter read value depends on the operation mode and how the counter was stopped:

- by a reset
- by a counter stop trigger (TAUJnTT.TAUJnTTm = 1)

The following table lists the initial counter read values after the counter has stopped (TAUJnTE.TAUJnTEm= 0) and re-enabled (TAUJnTS.TAUJnTSm = 1).

The table also contains the counter read value one count after the counter is enabled (TAUJnTS.TAUJnTSm = 1) for modes where the counter waits for a start trigger.

Table 17-62 TAUJnCNTm read values after the counter is re-enabled

Mode name	Count method (up/down)	TAUJnCNTm value		
		When operation mode is changed after reset	After stop trigger	After one count
Interval timer mode	Count down	FFFF FFFF$_H$	Stop value	-
Capture mode	Count up	0000 0000$_H$	Stop value	-
One-count mode	Count down	FFFF FFFF$_H$	Stop value	FFFF FFFF$_H$
Capture & one-count mode	Count up	0000 0000$_H$	Stop value	Captured value + 1 (TAUJnCDRm)
Count capture mode	Count up	0000 0000$_H$	Stop value	-
Gate count mode	Count down	FFFF FFFF$_H$	Stop value	Stop value
Capture & gate count mode	Count up	0000 0000$_H$	Stop value	Stop value

Note If the operation mode is changed while the counter is stopped, the initial counter value after counter restart is undefined. The operation mode is changed by the TAUJnCMORm.TAUJnMD[4:1] bits.

(3) TAUJnCMORm - TAUJn channel mode OS register

This register controls channel m operation.

Access This register can be read or written in 16-bit units. Writing is only possible while the counter is stopped (TAUJnTE.TAUJnTEm = 0).

Address <TAUJn_base> + 80_H + m x 4_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAUJn CKS[1:0]		TAUJn CCS[1:0]		TAUJn MAS	TAUJnSTS[2:0]			TAUJn COS[1:0]		-	TAUJnMD[4:0]				
R/W		R/W		R/W	R/W			R/W		R	R/W				

Table 17-63 TAUJnCMORm register contents (1/3)

Bit position	Bit name	Function
15, 14	TAUJn CKS[1:0]	Selects the prescaler output. The prescaler output is used for the TAUJnTTINm input edge detection circuit. It can also be used as the count clock depending on bits TAUJnCMORm.CCS[1:0]. TAUJn CKS1 / TAUJn CKS0 / Selected prescaler output 0 / 0 / CK0 0 / 1 / CK1 1 / 0 / CK2 1 / 1 / CK3
13, 12	TAUJn CCS[1:0]	Selects the count clock for TAUJnCNTm counter. TAUJn CCS1 / TAUJn CCS0 / Selected count clock 0 / 0 / Prescaler output specified by TAUJnCMORm.TAUJnCKS[1:0] 0 / 1 / Valid edge of TAUJnTTINm input signal 1 / 0 / Setting prohibited 1 / 1
11	TAUJnMAS	Specifies the channel as master or slave channel during synchronous channel operation. 0: Slave 1: Master This bit is only valid for even channels (CHm_even). For odd channels (CHm_odd), it is fixed to 0.

Table 17-63 TAUJnCMORm register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>10 to 8</td><td>TAUJn STS[2:0]</td><td>Selects the external start trigger.
<table>
<tr><th>TAUJn STS2</th><th>TAUJn STS1</th><th>TAUJn STS0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Software trigger</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Valid edge of the TAUJnTTINm input signal. TAUJnCMURm.TAUJnTIS[1:0] specifies the valid edge.</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Valid edge of the TAUJnTTINm input signal is used as the start trigger, and the reverse edge is used as the stop trigger.</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>INT of the master channel</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td rowspan="3">Setting prohibited</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table></td></tr>
<tr><td>7, 6</td><td>TAUJn COS[1:0]</td><td>Specifies when the capture register TAUJnCDRm and the overflow flag TAUJnCSRm.TAUJnOVF of channel m are updated.
These bits are only valid if channel m is in capture mode.
<table>
<tr><th>TAUJn COS1</th><th>TAUJn COS0</th><th>TAUJnCDRm</th><th>TAUJnCSRm.TAUJnOVF</th></tr>
<tr><td>0</td><td>0</td><td rowspan="2">Updated upon detection of a TAUJnTTINm input valid edge.</td><td>Updated (cleared or set) upon detection of a TAUJnTTINm input valid edge:
• If a counter overflow has occurred since the last valid edge detection, TAUJnCSRm.TAUJnOVF is set.
• If no counter overflow has occurred since the last valid edge detection, TAUJnCSRm.TAUJnOVF is cleared.</td></tr>
<tr><td>0</td><td>1</td><td>Set upon counter overflow and cleared by setting TAUJnCSCm.TAUJnCLOV.</td></tr>
<tr><td>1</td><td>0</td><td rowspan="2">Updated upon detection of a TAUJnTTINm input valid edge and upon counter overflow:
• TAUJnTTINm input valid edge: Counter value is written to TAUJnCDRm.
• Overflow: FFFF FFFF$_H$ is loaded to TAUJnCDRm.
The next TAUJnTTINm input valid edge detection is ignored.</td><td>Not set.</td></tr>
<tr><td>1</td><td>1</td><td>Set upon counter overflow and cleared by setting TAUJnCSCm.TAUJnCLOV.</td></tr>
</table></td></tr>
</table>

Table 17-63 TAUJnCMORm register contents (3/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>4 to 0</td><td>TAUJn MD[4:0]</td><td>Specifies the operation mode.
<table>
<tr><th>TAUJn MD4</th><th>TAUJn MD3</th><th>TAUJnMD2</th><th>TAUJnMD1</th><th>TAUJnMD0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1/0</td><td>Interval timer mode</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1/0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1/0</td><td>Capture mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>1/0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1/0</td><td>One-count mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>1/0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>Capture & one-count mode</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td rowspan="4">1/0</td><td rowspan="4">Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1/0</td><td>Count capture mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>Gate count mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>Capture & gate count mode</td></tr>
</table>
<table>
<tr><th>Mode</th><th>Role of the MD0 bit</th></tr>
<tr><td>Interval timer mode
Capture mode
Count capture mode</td><td>Specifies whether the INTTAUJnIm signal is output when the counter starts counting (when the start trigger is input).
0: Does not output INTTAUJnIm.
1: Outputs INTTAUJnIm</td></tr>
<tr><td>One-count mode
Gate count mode</td><td>Enables/disables start trigger detection during counting:
0: Disables
1: Enables</td></tr>
<tr><td>Capture & one-count mode
Capture & gate count mode</td><td>This bit must be set to 0.</td></tr>
</table>
</td></tr>
</table>

(4) TAUJnCMURm - TAUJn channel mode user register

This register specifies the type of valid edge detection used for the TAUJnTTINm input.

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + 20_H + m x 4_H

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	-	-	TAUJnTIS[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-64 TAUJnCMURm register contents

Bit position	Bit name	Function
1, 0	TAUJnTIS[1:0]	Specifies the valid edge of the TAUJnTTINm input. • Edge detection for TAUJnTTINm input signals is performed based on the prescaler output selected by TAUJnCMORm.TAUJnCKS[1:0].

TAUJn TIS1	TAUJn TIS0	Description
0	0	Falling edge
0	1	Rising edge
1	0	Rising and falling edges (low-width measurement selection). Start trigger: falling edge Stop trigger (capture): rising edge
1	1	Rising and falling edges (high-width measurement selection). Start trigger: rising edge Stop trigger (capture): falling edge

(5) TAUJnCSRm - TAUJn channel status register

This register indicates the overflow status of channel m.

Access This register can be read in 8-bit units.

Address <TAUJn_base> + 30_H + m x 4_H

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	TAUJnOVF
R	R	R	R	R	R	R	R

Table 17-65 TAUJnCSRm register contents

Bit position	Bit name	Function
0	TAUJnOVF	Indicates the counter overflow status: 0: No overflow occurred 1: Overflow occurred This bit is only used in the following modes: • Capture mode • Capture & one-count mode • Count capture mode • Capture & gate count mode The function of this bit depends on the setting of control bits TAUJnCMORm.TAUJnCOS[1:0].

(6) TAUJnCSCm - TAUJn channel status clear register

This register is a trigger register for clearing the overflow flag TAUJnCSRm.TAUJnOVF of a channel m.

Access This register can be written in 8-bit units. It is always read as 00_H.

Address <TAUJn_base> + 40_H + m x 4_H

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	-	-	0	TAUJnCLOV
R	R	R	R	R	R	R	W

Table 17-66 TAUJnCSCm register contents

Bit position	Bit name	Function
0	TAUJnCLOV	0: No function 1: Clears the overflow flag TAUJnCSRm.TAUJnOVF

(7) TAUJnTS - TAUJn channel start trigger register

This register enables the counter for each channel.

Access This register can be written in 8-bit units. It is always read as 00_H.

Address <TAUJn_base> + 54_H

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnTS03	TAUJnTS02	TAUJnTS01	TAUJnTS00
W	W	W	W	W	W	W	W

Table 17-67 TAUJnTS register contents

Bit position	Bit name	Function
3 to 0	TAUJnTSm	Enables the counter for channel m. 0: No function 1: Enables the counter and sets TAUJnTE.TAUJnTEm = 1. TAUJnTE.TAUJnTEm = 1 only *enables* counter. Whether the counter *starts* depends on the selected operation mode.

(8) TAUJnTE - TAUJn channel enable status register

This register indicates whether counter is enabled or disabled.

Access This register can be read in 8-bit units.

Address <TAUJn_base> + 50_H

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnTE03	TAUJnTE02	TAUJnTE01	TAUJnTE00
R	R	R	R	R	R	R	R

Table 17-68 TAUJnTE register contents

Bit position	Bit name	Function
3 to 0	TAUJnTEm	Indicates whether counter for channel m is enabled or disabled. 0: Counter disabled 1: Counter enabled This bit is set when TAUJnTSSTm (the synchronous channel start trigger signal) trigger input is detected or when TAUJnTS.TAUJnTSm is set. Setting TAUJnTT.TAUJnTTm to 1 resets this bit to 0.

(9) TAUJnTT - TAUJn channel stop trigger register

This register stops the counter for each channel.

Access This register can be written in 8-bit units. It is always read as 00_H.

Address <TAUJn_base> + 58_H

Initial Value 00_H.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnTT03	TAUJnTT02	TAUJnTT01	TAUJnTT00
W	W	W	W	W	W	W	W

Table 17-69 TAUJnTT register contents

Bit position	Bit name	Function
3 to 0	TAUJnTTm	Stops the counter of channel m. 0: No function 1: Stop the counter and reset TAUJnTE.TAUJnTEm. TAUJnCNTm, TAUJnTO.TAUJnTOm, and TAUJnTTOUTm retain their values from before the counter stopped.

17.19.4 TAUJn output register details

(1) TAUJnTOE - TAUJn channel output enable register

This register enables and disables independent channel output mode controlled by software.

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + 60_H

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnTOE03	TAUJnTOE02	TAUJnTOE01	TAUJnTOE00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-70 TAUJnTOE register contents

Bit position	Bit name	Function
3 to 0	TAUJnTOEm	Enables or disables the independent timer output function. 0: Disables the independent timer output function. 1: Enables the independent timer output function.

(2) TAUJnTOM - TAUJn channel output mode register

This register specifies the output mode of each channel.

Access This register can be read/written in 8-bit units. Writing is only possible while the counter is stopped (TAUJnTE.TAUJnTEm = 0).

Address <TAUJn_base> + 98_H

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnTOM03	TAUJnTOM02	TAUJnTOM01	TAUJnTOM00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-71 TAUJnTOM register contents

Bit position	Bit name	Function
3 to 0	TAUJnTOMm	Specifies the channel output mode. 0: Independent channel output mode 1: Synchronous channel output mode The output mode depends on several channel output control bits, as can be seen in *Table 17-11 “Channel output modes” on page 1189*.

(3) TAUJnTOC - TAUJn channel output configuration register

This register specifies the output mode of each channel in combination with TAUJnTOMm.

Access This register can be read/written in 8-bit units. Writing is only possible while the counter is stopped (TAUJnTE.TAUJnTEm = 0).

Address <TAUJn_base> + $9C_H$

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnTOC03	TAUJnTOC02	TAUJnTOC01	TAUJnTOC00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-72 TAUJnTOC register contents

Bit position	Bit name	Function
3 to 0	TAUJnTOCm	Specifies the output mode. 0: Operation mode 1 (toggle mode) 1: No function

17.19.5 TAUJn channel output level register details

(1) TAUJnTO - TAUJn channel output register

This register specifies and reads the level of TAUJnTTOUTm.

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + $5C_H$

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnTO03	TAUJnTO02	TAUJnTO01	TAUJnTO00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-73 TAUJnTO register contents

Bit position	Bit name	Function
3 to 0	TAUJnTOm	Specifies and reads the level of TAUJnTTOUTm. 0: Low level 1: High level Only TAUJnTOm bits for which the independent channel output function is disabled (TAUJnTOEm = 0) can be written.

(2) TAUJnTOL - TAUJn channel output level register

This register specifies the output logic of the channel output bit (TAUJnTO.TAUJnTOm).

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + 64_H

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnTOL03	TAUJnTOL02	TAUJnTOL01	TAUJnTOL00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-74 TAUJnTOL register contents

Bit position	Bit name	Function
3 to 0	TAUJnTOLm	Specifies the output logic of the channel m output bit (TAUJnTO.TAUJnTOm). 0: Positive logic (active high) 1: Inverted logic (active low)

17.19.6 TAUJn simultaneous rewrite register details

(1) TAUJnRDE - TAUJn channel reload data enable register

This register enables and disables simultaneous rewrite of the data register TAUJnCDRm.

Access This register can be read/written in 8-bit units. Writing is only possible while TAUJnTE.TAUJnTEm is 0.

Address <TAUJn_base> + $A0_H$

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnRDE03	TAUJnRDE02	TAUJnRDE01	TAUJnRDE00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-75 TAUJnRDE register contents

Bit position	Bit name	Function
3 to 0	TAUJnRDEm	Enables and disables simultaneous rewrite of the data register of channel m. 0: Disables simultaneous rewrite 1: Enabled simultaneous rewrite

(2) TAUJnRDM - TAUJn channel reload data mode register

This register selects the timing for generating a simultaneous rewrite control signal.

Access This register can be read/written in 8-bit units. Writing is only possible while TAUJnTE.TAUJnTEm is 0.

Address <TAUJn_base> + $A4_H$

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnRDM03	TAUJnRDM02	TAUJnRDM01	TAUJnRDM00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-76 TAUJnRDM register contents

Bit position	Bit name	Function
3 to 0	TAUJnRDMm	Selects the timing for generating a simultaneous rewrite control signal. 0: When the master channel counter starts counting 1: No function These bits only apply when TAUJnRDE.TAUJnRDEm = 1.

(3) TAUJnRDT - TAUJn channel reload data trigger register

This register triggers the simultaneous rewrite pending state.

Access This register can be written in 8-bit units. It is always read as 00_H.

Address <TAUJn_base> + 68_H

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnRDT03	TAUJnRDT02	TAUJnRDT01	TAUJnRDT00
W	W	W	W	W	W	W	W

Table 17-77 TAUJnRDT register contents

Bit position	Bit name	Function
3 to 0	TAUJnRDTm	Triggers the simultaneous rewrite pending state. 0: No function 1: Simultaneous rewrite pending state is triggered. The simultaneous rewrite pending flag (TAUJnRSFm) is set to 1. The system waits for the simultaneous rewrite trigger.

(4) TAUJnRSF - TAUJn channel reload status register

This flag register indicates the simultaneous rewriting status.

Access This register can be read in 8-bit units.

Address <TAUJn_base> + $6C_H$

Initial Value 00_H. This bit is initialized by any reset.

7	6	5	4	3	2	1	0
-	-	-	-	TAUJnRSF03	TAUJnRSF02	TAUJnRSF01	TAUJnRSF00
R	R	R	R	R	R	R	R

Table 17-78 TAUJnRSF register contents

Bit position	Bit name	Function
3 to 0	TAUJnRSFm	Indicates the simultaneous rewrite status. 0: Simultaneous rewriting was performed due to the generation of a simultaneous rewrite trigger. 1: Simultaneous rewriting is pending (TAUJnRDTm = 1).

Chapter 18 Real-Time Clock (RTCA)

This chapter describes the real-time clock (RTCA).

The first section describes all properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names. The subsequent sections describe the features that apply to all implementations.

18.1 V850E2/Sx4-H RTCA Features

Instances

This microcontroller has the following number of instances of RTCA:

Table 18-1 Instances of RTCA

RTCA	
Number of instances	1
Name	RTCA0

Instances index n

Throughout this chapter, the individual instances of RTCA are identified by “n” (n = 0), for example, RTCAnCTL0 for RTCAn control register 0.

Register addresses

All RTCAn register addresses are given as addresses offset from the individual base address <RTCAn_base>.
The base address <RTCAn_base> of each RTCAn is listed in the following table:

Table 18-2 Register base addresses <RTCAn_base>

RTCAn	<RTCAn_base> address
RTCA0	FF81 4000_H

Clock supply

The following clocks are supplied to RTCA:

Table 18-3 RTCAn clock supply

RTCAn	Clock	Connected to:
RTCA0	RTCATCKI	Clock generator RTCATCKI = 32 kHz up to 4.194304 MHz
	PCLK	Clock generator PCLK

Interrupts

RTCA can generate the following interrupt requests:

Table 18-4 RTCAn interrupt requests

RTCAn signals	Function	Connected to:
RTCATINT1S	1 second interval interrupt	Interrupt controller INTRTCA01S
RTCATINTAL	Alarm interrupt	Interrupt controller INTRTCA0AL
RTCATINTR	Fixed interval interrupt	Interrupt controller INTRTCA0R

RTCA hardware reset

RTCA and its registers are initialized by the following reset signal:

Table 18-5 RTCAn reset signal

RTCAn	Reset signal
RTCAn	System reset SYSRES

I/O signals

The I/O signals of RTCA are listed in the following table:

Table 18-6 RTCAn I/O signals

RTCAn signals	Function	Connected to:
RTCAT1HZ	1 second interval output	Port RTCA0OUT

18.2 Functional Overview

Features summary

The Real-Time Clock RTCA has the following features:

- Count clock selection from 32 kHz to 4.194304 MHz
- Counters for years, months, day of the month, day of the week, hours, minutes, seconds, and a sub-counter

  The calendar covers 99 years. Leap years are handled by hardware automatically.
- One Hz pulse output function
- Fixed interval interrupt function
- Alarm interrupt function
- Clock error correction function if a 32.768 kHz count clock is used

The block diagram shows the main components of the RTCA.

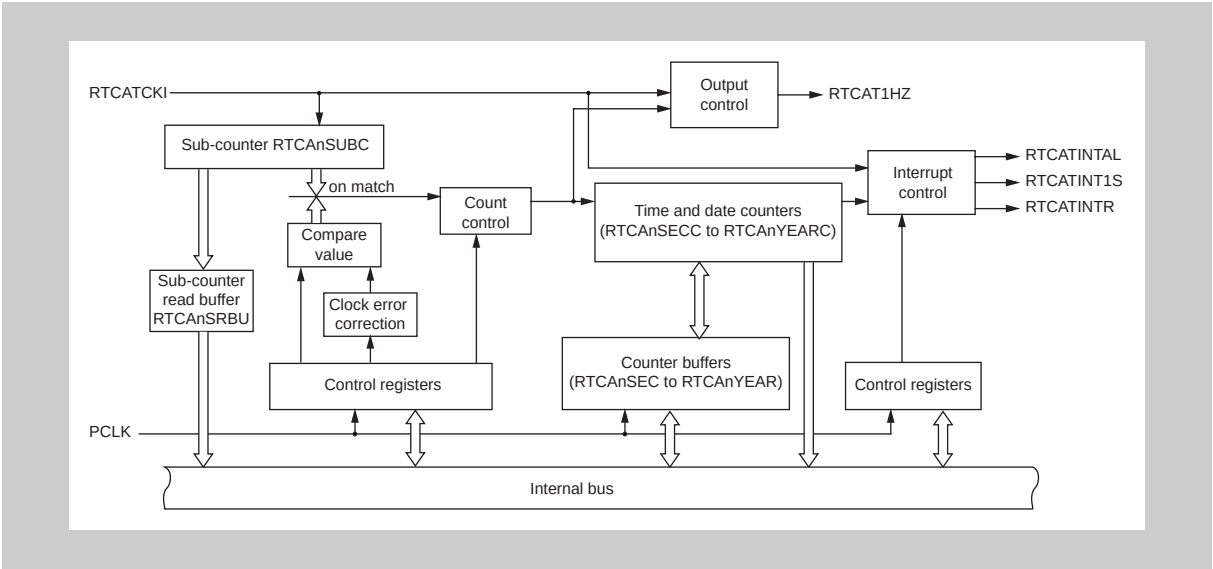


Figure 18-1 Block diagram of the RTCA

18.3 Functional Description

The Real-Time Clock RTCA provides information about the present time and date and can generate wake-up signals (interrupts, alarms). This information is derived from the count clock RTCATCKI.

Sub-counter

RTCATCKI is the input to the sub-counter RTCAnSUBC. The sub-counter counts up from zero until it reaches the compare value. The compare value is always defined as the frequency of RTCATCKI – 1 (in Hz). Thus, the sub-counter overflows after one second. It is then reset to zero and triggers the seconds counter RTCAnSECC (and, if desired, the interrupt RTCATINT1S).

The sub-counter can generate a fixed interval interrupt every 0.25 seconds, 0.5 seconds, or 1 second, and a 1 Hz output pulse.

Time and date counters

The counters for minutes, hours, day of the week, day of the month, months, and years also count up. They have their own overflow limits. One triggers the next one.

The overflow limit of the counter for the day of the month (RTCAnDAYC) depends on the present month (28, 30, or 31 days) and (in February) on the year counter RTCAnYEARC (years 0, 4, 8, 12, etc. are considered leap years).

The hours counter RTCAnHOURC can be switched between 12 and 24 hour format.

The counters for seconds, minutes, hours, day of the month, and month can generate a fixed interval interrupt upon overflow (RTCATINTR).

The counters for minutes, hours, and day of the week can also generate an alarm interrupt, e.g. every Tuesday and Thursday at 10:32 (RTCATINTAL).

Counter buffers

All counters can be read directly at any time. The clock signal used to access the read/write registers and the count clock are usually asychronous. An overflow of the sub-counter during the read operation can make all read values obsolete. Therefore, reading the counters must be performed using a special procedure. For details, see *18.5.3 “Reading clock counters” on page 1331*.

For reasons of synchronization, the counters cannot be written directly.

For reading and writing, all counters are accompanied by buffer registers. The buffer registers provide a synchronized way for reading the counters and for setting time and date. When they are used, the operation of the sub-counter must first be suspended and then re-activated (see *18.5.3 “Reading clock counters” on page 1331* and *18.5.2 “Updating clock counters” on page 1330*.

The RTCAnTIMEC and RTCAnCALC registers and their corresponding buffer registers can be used to access the time (hours, minutes and seconds) or the date (day of the week, day of the month, month, and year) with one read/write operation.

18.3.1 Operation modes

The RTCA provides two operation modes:

- Frequency selection mode
- 32.768 kHz mode

The operation mode that can be used depends on the available input clock RTCATCKI. The operation mode specifies the sub-counter compare value that is used to trigger the seconds counter and thus all subsequent counters. Clock error correction is only possible in 32.768 kHz mode.

The following table provides an overview of the properties of the two operation modes.

Table 18-7 RTCA operation mode overview

	Frequency selection mode	32.768 kHz mode	
		Clock correction disabled	Clock correction enabled
Allowed input clock RTCATCKI	Any frequency from 32 kHz to 4.194304 MHz	32.768 kHz	Any frequency from 32.76180000 kHz to 32.77420000 kHz
Sub-counter RTCAnSUBC operation	• Counter overflow at value of RTCAnSCMP • RTCAnSCMP must be set to RTCATCKI - 1 (in Hz)	Counter overflow at $7FFF_H$	Counter overflow at • $7FFF_H$ or • Every 20 or 60 seconds: $7FFF_H \pm$ RTCAnSUBU.RTCAnF[5:0]

The operation mode is selected by control bit RTCAnCTL0.RTCAnSLSB. For how to specify the operation mode during RTCA initialization, see *18.5.1 "Initial setting of the RTCA" on page 1328*.

Cautions

1. The input clock RTCATCKI must not be outside the allowed frequency range.
2. The operation mode must not be changed while sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1).

18.3.2 Clock counter format

The clock counters (RTCAnSECC to RTCAnYEARC) operate on binary coded decimals (BCD): Each digit is represented by its own binary sequence.

Depending on the valid data range, the number of bits for a digit differs. For example, the tens digit of the month of the year counter has only one bit (for 0 and 1) whereas the tens digit of the minutes counter has 3 digits (for 0 to 5).

The following table lists the decimals 0 to 60 in binary and BCD.

Table 18-8 Example of BCD Code – seconds or minutes counter (0 to 59)

Decimal	Binary	BCD
0	000000	000 0000
1	000001	000 0001

Table 18-8 Example of BCD Code – seconds or minutes counter (0 to 59)

Decimal	Binary	BCD
2	000010	000 0010
3	000011	000 0011
4	000100	000 0100
5	000101	000 0101
6	000110	000 0110
7	000111	000 0111
8	001000	000 1000
9	001001	000 1001
10	001010	001 0000
11	001011	001 0001
12	001100	001 0010
...	...	...
58	111010	101 1000
59	111011	101 1001

18.3.3 Fixed interval interrupt function

Interrupt RTCATINTR can be specified to occur after every 0.25 seconds, 0.5 seconds, 1 (full) second, 1 (full) minute, 1 (full) hour, or 1 (full) month.

The fixed interval interrupt function is controlled by bits RTCAnCTL1.RTCAnCT[2:0].

18.3.4 Alarm interrupt function

Interrupt RTCATINTAL can be specified to occur at a certain time on one or several days of the week. This interrupt can be used as a wake-up signal.

The alarm interrupt function is enabled by bit RTCAnCTL1.RTCAnALME.

The alarm setting is specified by the following control registers:

- RTCAnALW selects the weekday(s)

  The allocation of bits to weekdays is defined by the day of the week counter RTCAnWEEKC.

- RTCAnALH and RTCAnALM specify the hour and minute in BCD

Examples The following tables show some examplary settings of the alarm control registers for both 12 hour and 24 hour format.

In this example, Sunday is RTCAnWEEK = 0, Monday is RTCAnWEEK = 1, Tuesday is RTCAnWEEK = 2, …, Saturday is RTCAnWEEK = 6:

Table 18-9 Alarm setting in 12-hour format (RTCAnCTL0.RTCAnAMPM = 0)

Alarm setting time	RTCAnALW	RTCAnALH	RTCAnALM
Sunday 7:00 am	01_H	07_H	00_H
Sunday, Monday 12:15 pm	03_H	32_H	15_H
Monday, Wednesday, Friday 5:30 pm	26_H	25_H	30_H
Daily, 10:45 pm	$7F_H$	30_H	45_H

Table 18-10 Alarm setting in 24-hour format (RTCAnCTL0.RTCAnAMPM = 1)

Alarm setting time	RTCAnALW	RTCAnALH	RTCAnALM
Sunday 7:00	01_H	07_H	00_H
Sunday, Monday 12:15	03_H	12_H	15_H
Monday, Wednesday, Friday 17:30	26_H	17_H	30_H
Daily, 22:45	$7F_H$	22_H	45_H

18.3.5 Clock error correction

Clock error correction compensates for deviations of the oscillator from the nominal clock rate. With clock error correction input clock rates from 32.76180 kHz to 32.77420 kHz are possible.

The clock error correction function is only available in 32.768 kHz operation mode. In this operation mode, a nominal clock rate of 32.768 kHz is expected and the sub-counters overflow value is fixed to $7FFF_H$.

The following figures illustrate the clock error when the input clock rate deviates from the nominal clock.

RTCATCKI = 32.768 kHz

Figure 18-2 "RTCATCKI = 32.768 kHz, no clock error correction required" shows the timing diagram if RTCATCKI matches the nominal clock rate of 32.768 kHz. No clock error correction is required.

Counting from 0 to 32767 (0 to $7FFF_H$) with a 32.768 kHz clock is equal to 1 second exactly.

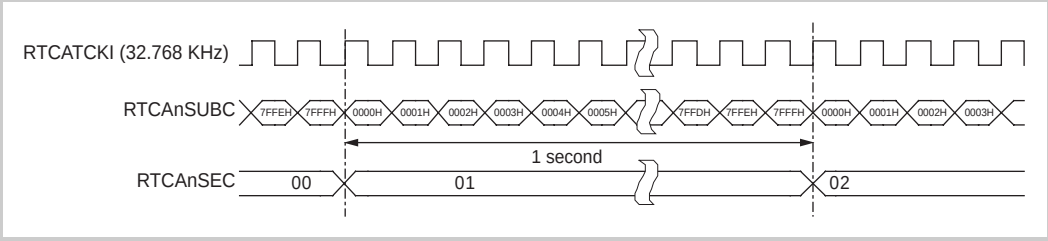


Figure 18-2 RTCATCKI = 32.768 kHz, no clock error correction required

RTCATCKI = 32.769 kHz

Figure 18-3 "RTCATCKI = 32.769 kHz, no clock error correction enabled" shows the timing diagram if RTCATCKI deviates from the nominal clock rate of 32.768 kHz. In this example, RTCATCKI is connected to a 32.769 kHz oscillator. Clock error correction is not enabled.

Counting from 0 to 32767 (0 to $7FFF_H$) with a 32.769 kHz clock is equal to approximately 0.99997 seconds (32768/32769). A "+ error" (faster than 32.768 kHz) occurs. In 1 month, RTCA deviates approximately -79 seconds from the real time.

Error = (32768/ 32769 - 1) x 60 (s) x 60 (min) x 24 (h) x 30 (d)

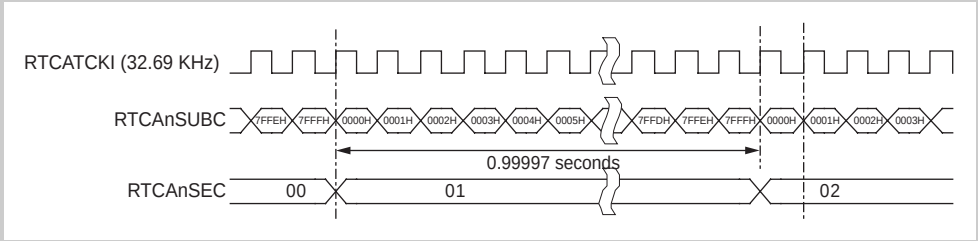


Figure 18-3 RTCATCKI = 32.769 kHz, no clock error correction enabled

Clock error correction is performed by stretching/reducing the 1 second period of the sub-counter in regular intervals. The sub-counter's upper limit of $7FFF_H$ is increased or decreased by setting the following parameters in register RTCAnSUBU:

- A correction value greater than one
- An operator (add/subtract)
- A repetition rate (20 or 60 seconds)

The corrected overflow value becomes effective every 20 or 60 seconds, so that on the average RTCAnSECC is triggered exactly every second.

(1) Setting the correction value and the operator

The correction value and operator are specified by the RTCAnF[6:0] bits of the RTCAnSUBU register:

- RTCAnF[6] specifies whether the overflow value is incremented or decremented
- RTCAnF[5:0] specifies the correction value

The correction values are calculated as follows:

Table 18-11 Correction value settings

RTCAnF[6]	Increment/decrement	Correction value
0	Increment	(Value of RTCAnF[5:0] – 1) x 2
1	Decrement	(Inverted value of RTCAnF[5:0] + 1) x 2

Some examples are given in the following table:

Table 18-12 Correction value examples

RTCAnF[6]	RTCAnF[5:0]	Correction value	Count limit of RTCAnSUBC
0	15_H	$(15_H - 1) \times 2 = 40$	32768 + 40 = 32808
1	15_H	$(\overline{15}_H + 1) \times 2$ $= (2A_H + 1) \times 2$ $= 86$	32768 – 86 = 32682

(2) Impact of the repetition rate

The correction value set by RTCAnF[6:0] does not change the count limit of RTCAnSUBC every second. The repetition rate at which the correction value becomes effective is specified by bit RTCAnDEV.

This bit also influences the size of the correctable frequency range and the correction accuracy.

The following table summarizes the RTCAnDEV settings.

Table 18-13 Setting of bit RTCAnSUBU.RTCAnDEV

RTCAnDEV	Count limit of RTCAnSUBC is changed	Frequency range that can be corrected	Correction accuracy
0	Every 20 seconds when RTCAnSECC = 00, 20, or 40	32.76180000 to 32.77420000 kHz	
1	Every 60 seconds when RTCAnSECC = 00	32.76593333 to 32.77006667 kHz	Three times higher than for RTCAnDEV = 0

(3) Sample settings

The frequencies that can be corrected, as well as the setting values of bits RTCAnDEV and RTCAnF[6:0], are listed in the following table.

Table 18-14 Correctable frequency range when RTCAnDEV = 0 (1/2)

Connected clock frequency	RTCAnF6	RTCAnF[5:0]	Correction value
–	0	000000	No correction
–	0	000001	No correction
32.76810000 kHz	0	000010	Once every 20 s, RTCAnSUBC count value + 2
32.76820000 kHz	0	000011	Once every 20 s, RTCAnSUBC count value + 4
32.76830000 kHz	0	000100	Once every 20 s, RTCAnSUBC count value + 6
...	...	...	...
32.77400000 kHz	0	111011	Once every 20 s, RTCAnSUBC count value + 120
32.77410000 kHz	0	111110	Once every 20 s, RTCAnSUBC count value + 122
32.77420000 kHz (upper limit)	0	111111	Once every 20 s, RTCAnSUBC count value + 124
–	1	000000	No correction
–	1	000001	No correction
32.76180000 kHz (lower limit)	1	000010	Once every 20 s, RTCAnSUBC count value – 124
32.76190000 kHz	1	000011	Once every 20 s, RTCAnSUBC count value – 122
32.76200000 kHz	1	000100	Once every 20 s, RTCAnSUBC count value – 120
...	...	...	...

Table 18-14 Correctable frequency range when RTCAnDEV = 0 (2/2)

Connected clock frequency	RTCAnF6	RTCAnF[5:0]	Correction value
32.76770000 Hz	1	11011	Once every 20 s, RTCAnSUBC count value – 6
32.76780000 kHz	1	11110	Once every 20 s, RTCAnSUBC count value – 4
32.76790000 kHz	1	11111	Once every 20 s, RTCAnSUBC count value – 2

Table 18-15 Correctable frequency range when RTCAnDEV = 1

Connected clock frequency	RTCAnF6	RTCAnF[5:0]	RTCAnSUBC correction value
–	0	000000	No correction
–	0	000001	No correction
32.76803333 kHz	0	000010	Once every 60 s, RTCAnSUBC count value + 2
32.76806667 kHz	0	000011	Once every 60 s, RTCAnSUBC count value + 4
32.76810000 kHz	0	000100	Once every 60 s, RTCAnSUBC count value + 6
...	...	...	...
32.77000000 kHz	0	111011	Once every 60 s, RTCAnSUBC count value + 120
32.77003333 kHz	0	111110	Once every 60 s, RTCAnSUBC count value + 122
32.77006667 kHz (upper limit)	0	111111	Once every 60 s, RTCAnSUBC count value + 124
–	1	000000	No correction
–	1	000001	No correction
32.76593333 kHz (lower limit)	1	000010	Once every 60 s, RTCAnSUBC count value – 124
32.76596667 kHz	1	000011	Once every 60 s, RTCAnSUBC count value – 122
32.76600000 kHz	1	000100	Once every 60 s, RTCAnSUBC count value – 120
...	...	...	...
32.76790000 kHz	1	11011	Once every 60 s, RTCAnSUBC count value – 6
32.76793333 kHz	1	11110	Once every 60 s, RTCAnSUBC count value – 4
32.76796667 kHz	1	11111	Once every 60 s, RTCAnSUBC count value – 2

18.4 Registers

This section contains a description of all RTCA registers.

18.4.1 RTCA register overview

The RTCA is controlled and operated by the following registers:

Table 18-16 RTCA register overview (1/2)

Register name	Symbol	Address
Control registers		
Control register 0	RTCAnCTL0	<RTCAn_base> + 00_H
Control register 1	RTCAnCTL1	<RTCAn_base> + 04_H
Control register 2	RTCAnCTL2	<RTCAn_base> + 08_H
Sub-counter registers		
Sub-count register	RTCAnSUBC	<RTCAn_base> + $0C_H$
Sub-count register read buffer	RTCAnSRBU	<RTCAn_base> + 10_H
Sub-counter compare register	RTCAnSCMP	<RTCAn_base> + $3C_H$
Clock error correction register	RTCAnSUBU	<RTCAn_base> + 38_H
Clock counter and buffer registers		
Seconds count register	RTCAnSECC	<RTCAn_base> + $4C_H$
Seconds count buffer register	RTCAnSEC	<RTCAn_base> + 14_H
Minute count register	RTCAnMINC	<RTCAn_base> + 50_H
Minute count buffer register	RTCAnMIN	<RTCAn_base> + 18_H
Hour count register	RTCAnHOURC	<RTCAn_base> + 54_H
Hour count buffer register	RTCAnHOUR	<RTCAn_base> + $1C_H$
Day of the week count register	RTCAnWEEKC	<RTCAn_base> + 58_H
Day of the week count buffer register	RTCAnWEEK	<RTCAn_base> + 20_H
Day count register	RTCAnDAYC	<RTCAn_base> + $5C_H$
Day count buffer register	RTCAnDAY	<RTCAn_base> + 24_H
Month count register	RTCAnMONC	<RTCAn_base> + 60_H
Month count buffer register	RTCAnMONTH	<RTCAn_base> + 28_H
Year count register	RTCAnYEARC	<RTCAn_base> + 64_H
Year count buffer register	RTCAnYEAR	<RTCAn_base> + $2C_H$
Special counter and buffer registers		
Time count register	RTCAnTIMEC	<RTCAn_base> + 68_H
Time count buffer register	RTCAnTIME	<RTCAn_base> + 30_H
Calendar count register	RTCAnCALC	<RTCAn_base> + $6C_H$
Calendar count buffer register	RTCAnCAL	<RTCAn_base> + 34_H

Table 18-16 RTCA register overview (2/2)

Register name	Symbol	Address
Alarm setting registers		
Alarm minute setting register	RTCAnALM	<RTCAn_base> + 40_H
Alarm hour setting register	RTCAnALH	<RTCAn_base> + 44_H
Alarm day of the week setting register	RTCAnALW	<RTCAn_base> + 48_H

<RTCAn_base> The base addresses <RTCAn_base> of the RTCAn is defined in the first section of this chapter under the key word “Register addresses”.

18.4.2 RTCA control register details

(1) RTCAnCTL0 – RTCA control register 0

This register controls the count operation of the sub-counter RTCAnSUBC, the format of the hours counter RTCAnHOURC and the alarm hour setting register RTCAnALH, and the operation mode.

Access This register can be read or written in 8-bit or 1-bit units.

Address <RTCAn_base>

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
RTCAnCE	RTCAnCEST	RTCAnAMPM	RTCAnSLSB	0	0	0	0
R/W	R	R/W	R/W	R	R	R	R

Table 18-17 RTCAnCTL0 register contents

Bit position	Bit name	Function
7	RTCAnCE	Starts/stops the sub-counter RTCAnSUBC operation. 0: Stops the sub-counter operation All output pins and all status flags in control register RTCAnCTL2 are cleared. 1: Starts the sub-counter operation The sub-counter counts up.
6	RTCAnCEST	Indicates whether the sub-counter is enabled: 0: Sub-counter operation disabled 1: Sub-counter operation enabled For how to use this status flag, see *18.5.1 "Initial setting of the RTCA" on page 1328*.
5	RTCAnAMPM	Selects the format of the hours counter RTCAnHOURC and the alarm hour setting register RTCAnALH: 0: 12 hour format (1 to 12, am/pm) 1: 24 hour format (0 to 23, military time) For details about the format, see the description of the hours counter *(6) "RTCAnHOUR – RTCA hour count buffer register"*.
4	RTCAnSLSB	Selects the operation mode: 0: 32.768 kHz mode 1: Frequency selection mode For details about the operation modes, see *18.3.1 "Operation modes" on page 1289*. The operation mode must not be changed while sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1). For details about the initialization of RTCAn, see *18.5.1 "Initial setting of the RTCA" on page 1328*.

(2) RTCAnCTL1 – RTCA control register 1

This register controls the interrupt request generation and the 1 Hz pulse output.

Access This register can be read or written in 8-bit or 1-bit units.

Address <RTCAn_base> + 04_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	RTCAn1HZE	RTCAnALME	RTCAn1SE	RTCAnCT2	RTCAnCT1	RTCAnCT0
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-18 RTCAnCTL1 register contents

Bit position	Bit name	Function
5	RTCAn1HZE	Enables/stops 1 Hz pulse output (RTCAT1HZ): 0: RTCAT1HZ disabled (RTCAT1HZ is fixed to 0) 1: RTCAT1HZ enabled
4	RTCAnALME	Enables/disables alarm interrupt request generation (RTCATINTAL): 0: RTCATINTAL disabled 1: RTCATINTAL enabled
3	RTCAn1SE	Enables/disables 1 second interrupt request generation (RTCATINT1S): 0: RTCATINT1S disabled 1: RTCATINT1S enabled
2 to 0	RTCAnCT[2:0]	Specifies the fixed interval interrupt request (RTCATINTR) setting:

RTCAnCT[2:0]	RTCATINTR interrupt request generation	
	Interval	Timing
000	No interrupt request generation	
001	Every 0.25 seconds	Every 0.25, 0.5, 0.75 and 1 second
010	Every 0.5 seconds	Every 0.5 and 1 second
011	Every second	Every 1 second
100	Every minute	Every 1 minute 00 seconds
101	Every hour	Every 1 hour 0 minutes 0 seconds
110	Every day	Every 1 day 0 hours 0 minutes 0 seconds (i.e. every midnight)
111	Every month	Every 1 month first day 0 hours 0 minutes 0 seconds (i.e. every first midnight of a month)

If the settings of RTCAnCT[2:0] are changed while sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), a glitch may be output to RTCATINTR. Implement appropriate interrupt mask processing procedures.

(3) RTCAnCTL2 – RTCA control register 2

This register contains status information and controls the data transfer from the sub-counter RTCAnSUBC to the dedicated sub-counter read buffer RTCAnSRBU and the operation setting of the clock counters (RTCAnSECC to RTCAnYEARC).

Access This register can be read or written in 8-bit or 1-bit units.

Address <RTCAn_base> + 08_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	RTCAnWUST	RTCAnWSST	RTCAnRSST	RTCAnRSUB	RTCAnWST	RTCAnWAIT
R	R	R	R	R	R/W	R	R/W

Table 18-19 RTCAnCTL2 register contents (1/2)

Bit position	Bit name	Function
5	RTCAnWUST	Indicates whether RTCAnSUBU write operation has been completed: 0: RTCAnSUBU write completed 1: RTCAnSUBU write in progress The write operation ends with the next sub-counter overflow. This bit is only valid when sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1). See *18.5.5 "Writing to RTCAnSUBU" on page 1335* for details.
4	RTCAnWSST	Indicates whether RTCAnSCMP write operation has been completed: 0: RTCAnSCMP write completed 1: RTCAnSCMP write in progress The write operation ends with the next sub-counter overflow. This bit is only valid when sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1). See *18.5.6 "Writing to RTCAnSCMP" on page 1336* for details.
3	RTCAnRSST	Indicates whether the value of the sub-counter (RTCAnSUBC) has been transferred to the sub-count register read buffer (RTCAnSRBU): 0: Transfer in progress 1: Transfer completed This bit is cleared (transfer is triggered) by RTCAnRSUB = 1. This bit is automatically set when the transfer is completed. See *18.5.4 "Reading RTCAnSRBU" on page 1334* for details.

Table 18-19 RTCAnCTL2 register contents (2/2)

Bit position	Bit name	Function
2	RTCAnRSUB	Enables/disables the transfer of the value of the sub-counter (RTCAnSUBC) to the dedicated read buffer (RTCAnSRBU): 0: Disables the transfer 1: Enables the transfer This bit is only valid when sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1). For details, see *18.6.3 "Timing of sub-counter buffer read while counter is enabled" on page 1339*.
1	RTCAnWST	Indicates the status of all clock counters (RTCAnSECC to RTCAnYEARC): 0: All clock counters are running 1: All clock counters are stopped The sub-counter is still running. The clock counters must be stopped before reading or writing clock counter values during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1). To stop the clock counters, set RTCAnWAIT = 1.
0	RTCAnWAIT	Restarts/stops all clock counters (RTCAnSECC to RTCAnYEARC): 0: Restarts all clock counters either immediatelly or immediatelly after the clock counter write operation finishes. 1: Stops all clock counters temporarily The sub-counter is still running. The clock counters must be stopped before reading or writing counter buffers during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1).

18.4.3 RTCA sub-counter register details

(1) RTCAnSUBC – RTCA sub-count register

This counter counts the 1 second reference time. It operates using the count clock RTCATCKI.

Access This register can be read in 32-bit units.

Address <RTCAn_base> + $0C_H$

<R> **Initial Value** $0000\ 0000_H$.

This register is initialized

- By any reset
- Upon write to the seconds count buffer (RTCAnSEC) or to the clock time setting register (RTCAnTIME).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	RTCAnSUBC[21:16]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnSUBC[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 18-20 RTCAnSUBC register contents

Bit position	Bit name	Function
21 to 0	RTCAnSUBC [21:0]	Sub-counter value. The sub-counter only operates while RTCAnCTL0.RTCAnCEST = 1.

Notes

1. This sub-counter operates with RTCATCKI while the read operation is clocked by PCLK. Reading this sub-counter during operation (RTCAnCTL0.RTCAnCEST = 1) is asynchronous to RTCATCKI and can lead to wrong results.

   Use the sub-count register read buffer (RTCAnSRBU) to read the sub-counter value during operation.

   For details, see *18.5.4 “Reading RTCAnSRBU” on page 1334*.

2. The count-operation of this sub-counter depends on the selected operation mode. See *18.3.1 “Operation modes” on page 1289* for details.

(2) RTCAnSRBU – RTCA sub-count register read buffer

This register is the read buffer for the sub-counter RTCAnSUBC.

Access This register can be read in 32-bit units.

Address <RTCAn_base> + 10_H

<R> **Initial Value** 0000 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	RTCAnSRBU[21:16]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnSRBU[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 18-21 RTCAnSRBU register contents

Bit position	Bit name	Function
21 to 0	RTCAnSRBU [21:0]	Sub-counter value at the time of the last RTCAnSUBC read. The sub-counter value is only read when control bit RTCAnCTL2.RTCAnRSUB is set to 1. The read operation is synchronized with RTCATCKI.

Note Perform RTCAnSRBU read according to the flow described in *18.5.4 “Reading RTCAnSRBU” on page 1334*.

(3) RTCAnSUBU – RTCA clock error correction register

This register enables and specifies clock error correction. This register only applies in 32.678 kHz mode (RTCAnCTL0.RTCAnSLSB = 0).

For details about clock error correction, see *18.3.5 "Clock error correction" on page 1292*.

Access This register can be read or written in 8-bit units.

Note the following when writing this register during sub-counter operation:

- Previous RTCAnSUBU write must be completed (RTCAnCTL2.RTCAnWUST = 0).
- The write operation ends with the next sub-counter overflow.

Address <RTCAn_base> + 38_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
RTCAnDEV	RTCAnF6	RTCAnF[5:0]					
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-22 RTCAnSUBU register contents

Bit position	Bit name	Function
7	RTCAnDEV	Specifies how often clock error correction is performed per minute: 0: Three times every minute (when RTCAnSECC equals 00, 20, and 40) 1: Once every minute (when RTCAnSECC equals 00)
6	RTCAnF6	Specifies whether the sub-counter value is incremented or decremented: 0: Incremented (+ correction) Incrementation value = (RTCAnF[5:0] value – 1) x 2 1: Decremented (– correction) Decrementation value = (inverted data of RTCAnF[5:0] value + 1) x 2
5 to 0	RTCAnF[5:0]	Error correction value.

Notes

1. When RTCAnF[5:0] = 00 0000_B, clock error correction is not performed.
2. Perform RTCAnSUBU write as described in
   - *18.5.1 "Initial setting of the RTCA" on page 1328* and
   - *18.5.5 "Writing to RTCAnSUBU" on page 1335*.

(4) RTCAnSCMP – RTCA sub-counter compare register

This register sets the compare value of the sub-counter RTCAnSUBC in frequency selection mode (RTCAnCTL0.RTCAnSLSB = 1).

When the sub-counter values matches the value of this register an overflow signal is output to the seconds counter RTCAnSECC and the sub-counter is cleared.

Set the value for this register according to the frequency of the input clock RTCATCKI.

Access This register can be read or written in 32-bit units.

Note the following when writing this register during sub-counter operation:

- Previous RTCAnSCMP write must be completed (RTCAnCTL2.RTCAnWSST = 0)
- The write operation ends with the next sub-counter overflow.

Address <RTCAn_base> + $3C_H$

<R> **Initial Value** $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	RTCAnSCMP[21:16]					
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnSCMP[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-23 RTCAnSCMP register contents

Bit position	Bit name	Function
21 to 0	RTCAnSCMP [21:0]	Sub-counter compare value in frequency selection mode.

Example The following example illustrates the correct setting of RTCAnSCMP:

- RTCATCKI = 4 MHz = 4,000,000 Hz
- Set RTCAnSCMP = 4,000,000 – 1 = 3,999,999 (decimal code) = $3D08FF_H$
- The seconds counter RTCAnSECC is triggered when the sub-counter value changes from $3D08FF_H$ to 0_H.

Notes

1. The operation of the RTCA cannot be guaranteed if a value of 3198 (decimal code) or lower is set to this register.
2. Perform RTCAnSCMP write as described in *18.5.1 “Initial setting of the RTCA” on page 1328* and *18.5.6 “Writing to RTCAnSCMP” on page 1336*.

18.4.4 RTCA clock counter and buffer register details

(1) RTCAnSECC – RTCA seconds count register

This register is the seconds counter. It counts seconds from 00 to 59 in BCD.

It has the following trigger properties:

- It is triggered by every overflow of the sub-counter RTCAnSUBC.

  If the sub-counter overflows while the seconds counter is stopped (RTCAnCTL2.RTCAnWST = 1), the seconds counter behaves as follows:

  – If *one* sub-counter overflow occurs while the seconds counter is stopped, the overflow is held internally.
  The seconds counter is incremented by one when it is restarted.

  – If *two or more* overflows occurs while the seconds counter is stopped, the overflow count cannot be held internally.
  The seconds counter is incremented by one when it is restarted.

  – If the seconds counter was updated while the seconds counter is stopped, the sub-counter overflow(s) are ignored.

- It outputs an overflow signal when the value changes from 59 to 00. The overflow signal triggers the minutes counter (RTCAnMINC).

Access This register can be read in 8-bit units.

Address <RTCAn_base> + $4C_H$

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	RTCAnSECC[6:0]						
R	R	R	R	R	R	R	R

Table 18-24 RTCAnSECC register contents

Bit position	Bit name	Function
6 to 0	RTCAnSECC [6:0]	Seconds in BCD.

Notes

1. Perform RTCAnSECC read according to the flow described in *18.5.3 “Reading clock counters” on page 1331*.
2. A start value can be assigned to this register by writing to the seconds count buffer register RTCAnSEC or to the clock time setting register RTCAnTIME. See
   - *18.5.1 “Initial setting of the RTCA” on page 1328* and
   - *18.5.2 “Updating clock counters” on page 1330*.

(2) RTCAnSEC – RTCA seconds count buffer register

This register is a buffer register to read/write the seconds counter RTCAnSECC.

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + 14_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	RTCAnSEC[6:0]						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-25 RTCAnSEC register contents

Bit position	Bit name	Function
6 to 0	RTCAnSEC [6:0]	Seconds in BCD.

Notes

1. When writing this register, only decimal values between 00 and 59 in BCD are allowed.
2. Perform RTCAnMIN read/write as described in
   - *18.5.1 “Initial setting of the RTCA” on page 1328*,
   - *18.5.2 “Updating clock counters” on page 1330*, and
   - *18.5.3 “Reading clock counters” on page 1331*.

(3) RTCAnMINC – RTCA minute count register

This register is the minutes counter. It counts minutes from 00 to 59 in BCD.

It has the following trigger properties:

- It is triggered by every overflow of the seconds counter RTCAnSECC.
- It outputs an overflow signal when the value changes from 59 to 00. The overflow signal triggers the hours counter (RTCAnHOURC).

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 50_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	RTCAnMINC[6:0]						
R	R	R	R	R	R	R	R

Table 18-26 RTCAnMINC register contents

Bit position	Bit name	Function
6 to 0	RTCAnMINC [6:0]	Minutes in BCD.

Notes

1. Perform RTCAnMINC read according to the flow described in *18.5.3 “Reading clock counters” on page 1331*.
2. A start value can be assigned to this register by writing to the minutes count buffer register RTCAnMIN or to the clock time setting register RTCAnTIME. See
   - *18.5.1 “Initial setting of the RTCA” on page 1328* and
   - *18.5.2 “Updating clock counters” on page 1330*.

(4) RTCAnMIN – RTCA minute count buffer register

This register is a buffer register to read/write the minutes counter RTCAnMINC.

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + 18_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	RTCAnMIN[6:0]						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-27 RTCAnMIN register contents

Bit position	Bit name	Function
6 to 0	RTCAnMIN [6:0]	Minutes in BCD.

Notes

1. When writing this register, only decimal values between 00 and 59 in BCD are allowed.
2. Perform RTCAnSEC read/write as described in
   - *18.5.1 “Initial setting of the RTCA” on page 1328*,
   - *18.5.2 “Updating clock counters” on page 1330*, and
   - *18.5.3 “Reading clock counters” on page 1331*.

(5) RTCAnHOURC – RTCA hour count register

This register is the hours counter. It counts the hours in BCD. The count range depends on the selected hour format. See *Table 18-29 "12 and 24 hour format" on page 1311*.

This register has the following trigger properties:

- It is triggered by every overflow of the minutes counter RTCAnMINC.
- It outputs an overflow signal when the value changes from 23 to 00 (in 24 hour format) or from 32 to 01 (in 12 hour format). The overflow signal triggers two counters:
  - Day of the week counter (RTCAnWEEKC)
  - Day of the month counter (RTCAnDAYC)

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 54_H

Initial Value 12_H when RTCAnCTL0.RTCAnAMPM = 0
00_H when RTCAnCTL0.RTCAnAMPM = 1.

This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	RTCAnHOURC[5:0]					
R	R	R	R	R	R	R	R

Table 18-28 RTCAnHOURC register contents

Bit position	Bit name	Function
5 to 0	RTCAnHOURC [5:0]	Hours in BCD. See *Table 18-29 "12 and 24 hour format" on page 1311* for details.

Notes

1. Perform RTCAnHOURC read according to the flow described in *18.5.3 "Reading clock counters" on page 1331*.
2. A start value can be assigned to this register by writing to the hours count buffer register RTCAnHOUR or to the clock time setting register RTCAnTIME. See
   - *18.5.1 "Initial setting of the RTCA" on page 1328* and
   - *18.5.2 "Updating clock counters" on page 1330*.

12 or 24 hour format

The count values of RTCAnHOURC depend on the selected hour format.

If 12 hour format is selected (RTCAnCTL0.RTCAnAMPM = 0), bit RTCAnHOURC5 is the am/ pm indicator:

- RTCAnHOURC5 = 0: am
- RTCAnHOURC5 = 1: pm

The following table shows the count range of RTCAnHOURC in both 12 and 24 hour format.

Table 18-29 12 and 24 hour format

12 hour format (RTCAnAMPM = 0)			24 hour format (RTCAnAMPM = 1)	
Time	**RTCAnHOUR**		**Time**	**RTCAnHOUR**
0 am	12_H		0	00_H
1 am	01_H		1	01_H
2 am	02_H		2	02_H
3 am	03_H		3	03_H
4 am	04_H		4	04_H
5 am	05_H		5	05_H
6 am	06_H		6	06_H
7 am	07_H		7	07_H
8 am	08_H		8	08_H
9 am	09_H		9	09_H
10 am	10_H		10	10_H
11 am	11_H		11	11_H
0 pm	32_H	⇓	12	12_H
1 pm	21_H		13	13_H
2 pm	22_H	pm indicator in 12 hour format: RTCAnHOURC5 = 1	14	14_H
3 pm	23_H		15	15_H
4 pm	24_H		16	16_H
5 pm	25_H		17	17_H
6 pm	26_H		18	18_H
7 pm	27_H		19	19_H
8 pm	28_H		20	20_H
9 pm	29_H		21	21_H
10 pm	30_H		22	22_H
11 pm	31_H		23	23_H

(6) RTCAnHOUR – RTCA hour count buffer register

This register is a buffer register to read/write the hours counter RTCAnHOURC.

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + $1C_H$

Initial Value 12_H when RTCAnCTL0.RTCAnAMPM = 0
00_H when RTCAnCTL0.RTCAnAMPM = 1.

This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	RTCAnHOUR[5:0]					
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-30 RTCAnHOUR register contents

Bit position	Bit name	Function
5 to 0	RTCAnHOUR [5:0]	Hours in BCD. See *Table 18-29 "12 and 24 hour format" on page 1311*for details.

Notes

1. When writing this register, only the following decimal values in BCD are allowed:
   - 12 hour format (RTCAnCTL0.RTCAnAMPM = 0):
     01 to 12 or 21 to 32
   - 24 hour format (RTCAnCTL0.RTCAnAMPM = 1):
     00 to 23
2. Perform RTCAnHOUR read/write as described in
   - *18.5.1 "Initial setting of the RTCA" on page 1328*,
   - *18.5.2 "Updating clock counters" on page 1330*, and
   - *18.5.3 "Reading clock counters" on page 1331*.

(7) RTCAnWEEKC – RTCA day of the week count register

This register is the day of the week counter. It counts from 0 to 6.

It has the following trigger properties:

- It is triggered by every overflow of the hours counter RTCAnHOURC.
- It does not trigger any other counter.

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 58_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	RTCAnWEEKC[2:0]		
R	R	R	R	R	R	R	R

Table 18-31 RTCAnWEEKC register contents

Bit position	Bit name	Function
2 to 0	RTCAnWEEKC [2:0]	Day of the week.

Notes

1. Perform RTCAnWEEKC read according to the flow described in *18.5.3 “Reading clock counters” on page 1331*.
2. A start value can be assigned to this register by writing to the week count buffer register RTCAnWEEK or to the calender setting register RTCAnCAL. See
   - *18.5.1 “Initial setting of the RTCA” on page 1328* and
   - *18.5.2 “Updating clock counters” on page 1330*.

(8) RTCAnWEEK – RTCA day of the week count buffer register

This register is a buffer register to read/write the day of the week counter RTCAnWEEKC.

There is no particular correspondence between the value of RTCAnWEEK and the day of the week. Set the correspondence according to the application to be used.

Example: 0 = Sunday, 1 = Monday, …, 6 = Saturday

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + 20_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	RTCAnWEEK[2:0]		
R	R	R	R	R	R/W	R/W	R/W

Table 18-32 RTCAnWEEK register contents

Bit position	Bit name	Function
2 to 0	RTCAnWEEK [2:0]	Day of the week.

Notes

1. When writing this register, only decimal values between 0 and 6 in BCD are allowed.
2. Perform RTCAnWEEK read/write as described in
   - *18.5.1 “Initial setting of the RTCA” on page 1328*,
   - *18.5.2 “Updating clock counters” on page 1330*, and
   - *18.5.3 “Reading clock counters” on page 1331*.

(9) RTCAnDAYC – RTCA day of the month count register

This register is the day of the month counter. It counts from 00 to a maximum of 31 in BCD, depending on the value of the month counter (RTCAnMONC) and the year counter (RTCAnYEARC):

- 01 to 31 (January, March, May, July, August, October, December)
- 01 to 30 (April, June, September, November)
- 01 to 29 (February, leap year)
- 01 to 28 (February, non-leap year)

Years 0, 4, 8, 12, etc. are considered leap years.

It has the following trigger properties:

- It is triggered by every overflow of the hours of the day counter RTCAnHOURC.
- It outputs an overflow signal when the value changes from 28, 29, 30, or 31 to 00, depending on the current month and year. The overflow signal triggers the month counter (RTCAnMONC).

Access This register can be read in 8-bit units.

Address <RTCAn_base> + $5C_H$

Initial Value 01_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	RTCAnDAYC[5:0]					
R	R	R	R	R	R	R	R

Table 18-33 RTCAnDAYC register contents

Bit position	Bit name	Function
5 to 0	RTCAnDAYC [5:0]	Day of the month in BCD

Notes

1. Perform RTCAnDAYC read according to the flow described in *18.5.3 “Reading clock counters” on page 1331*.
2. A start value can be assigned to this register by writing to the day count buffer register RTCAnDAY or to the calender setting register RTCAnCAL. See
   - *18.5.1 “Initial setting of the RTCA” on page 1328* and
   - *18.5.2 “Updating clock counters” on page 1330*.

(10) RTCAnDAY – RTCA day of the month count buffer register

This register is a buffer register to read/write the day of the month counter RTCAnDAYC.

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + 24_H

Initial Value 01_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	RTCAnDAY[5:0]					
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-34 RTCAnDAY register contents

Bit position	Bit name	Function
5 to 0	RTCAnDAY [5:0]	Day of the month in BCD.

Notes

1. When writing this register, only decimal values between 00 and 31 in BCD are allowed:
   - 01 to 31 (January, March, May, July, August, October, December)
   - 01 to 30 (April, June, September, November)
   - 01 to 29 (February, leap year)
   - 01 to 28 (February, non-leap year)
2. Perform RTCAnDAY read/write as described in
   - *18.5.1 “Initial setting of the RTCA” on page 1328*
   - *18.5.2 “Updating clock counters” on page 1330*
   - *18.5.3 “Reading clock counters” on page 1331*.

(11) RTCAnMONC – RTCA month count register

This register is the month counter. It counts the month of the year, starting from 01 to 12 in BCD.

It has the following trigger properties:

- It is triggered by every overflow of the sub-counter RTCAnDAYC.
- It outputs an overflow signal when the value changes from 12 to 01. The overflow signal triggers the year counter (RTCAnYEARC).

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 60_H

Initial Value 01_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	RTCAnMONC[4:0]				
R	R	R	R	R	R	R	R

Table 18-35 RTCAnMONC register contents

Bit position	Bit name	Function
4 to 0	RTCAnMONC [4:0]	Month of the year in BCD.

Notes

1. Perform RTCAnMONC read according to the flow described in *18.5.3 “Reading clock counters” on page 1331*.
2. A start value can be assigned to this register by writing to the month count buffer register RTCAnMONTH or to the calender setting register RTCAnCAL. See
   - *18.5.1 “Initial setting of the RTCA” on page 1328*
   - *18.5.2 “Updating clock counters” on page 1330*.

(12) RTCAnMONTH – RTCA month count buffer register

This register is a buffer register to read/write the month counter RTCAnMONC.

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + 28_H

Initial Value 01_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	RTCAnMONTH[4:0]				
R	R	R	R/W	R/W	R/W	R/W	R/W

Table 18-36 RTCAnMONTH register contents

Bit position	Bit name	Function
4 to 0	RTCAnMONTH [4:0]	Month of the year in BCD.

Notes

1. When writing this register, only decimal values between 01 and 12 in BCD are allowed.
2. Perform RTCAnMONTH read/write as described in
   - *18.5.1 “Initial setting of the RTCA” on page 1328*,
   - *18.5.2 “Updating clock counters” on page 1330*, and
   - *18.5.3 “Reading clock counters” on page 1331*.

(13) RTCAnYEARC – RTCA year count register

This register is the year counter. It counts years from 00 to a maximum of 99 in BCD.

Years 00, 04, 08, …, 92, and 96 (every four years) are considered leap years.

It has the following trigger properties:

- It is triggered by every overflow of the month counter RTCAnMONC.
- It does not trigger any other counter.

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 64_H

Initial Value 00_H. This register is initialized by any reset.

<R>

7	6	5	4	3	2	1	0
RTCAnYEAR[7:0]							
R	R	R	R	R	R	R	R

Table 18-37 RTCAnYEARC register contents

Bit position	Bit name	Function
7 to 0	RTCAnYEARC [7:0]	Year in BCD.

Notes

1. Perform RTCAnYEARC read according to the flow described in *18.5.3 “Reading clock counters” on page 1331*.
2. A start value can be assigned to this register by writing to the year count buffer register RTCAnYEAR or to the calender setting register RTCAnCAL. See
   - *18.5.1 “Initial setting of the RTCA” on page 1328* and
   - *18.5.2 “Updating clock counters” on page 1330*.

(14) RTCAnYEAR – RTCA year count buffer register

This register is a buffer register to read/write the year counter RTCAnYEARC.

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + $2C_H$

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
RTCAnYEAR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-38 RTCAnYEAR register contents

Bit position	Bit name	Function
7 to 0	RTCAnYEAR [7:0]	Year in BCD.

Notes

1. When writing this register, only decimal values between 00 and 99 in BCD are allowed.
2. Perform RTCAnYEAR read/write as described in
   - *18.5.1 “Initial setting of the RTCA” on page 1328*,
   - *18.5.2 “Updating clock counters” on page 1330*, and
   - *18.5.3 “Reading clock counters” on page 1331*.

18.4.5 RTCA special counter and buffer register details

(1) RTCAnTIMEC – RTCA clock time reading register

This register enables the RTCAnHOURC, RTCAnMINC, and RTCAnSECC counters to be read simultaneously.

Access This register can be read in 32-bit units.

Address <RTCAn_base> + 68_H

Initial Value $0012\ 0000_H$ when RTCAnCTL0.RTCAnAMPM = 0,
$0000\ 0000_H$ when RTCAnCTL0.RTCAnAMPM = 1.

This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	RTCAnHOURC[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnMINC[7:0]								RTCAnSECC[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 18-39 RTCAnTIMEC register contents

Bit position	Bit name	Function
23 to 16	RTCAnHOURC [7:0]	Hours in BCD. See *Table 18-29 "12 and 24 hour format" on page 1311* for details.
15 to 8	RTCAnMINC [7:0]	Minutes in BCD.
7 to 0	RTCAnSECC [7:0]	Seconds in BCD.

Notes

1. Perform RTCAnTIMEC read according to the flow described in *18.5.3 "Reading clock counters" on page 1331*.
2. A start value can be assigned to this register by writing to the clock time setting register RTCAnTIME. See
   - *18.5.1 "Initial setting of the RTCA" on page 1328* and
   - *18.5.2 "Updating clock counters" on page 1330*.

(2) RTCAnTIME – RTCA clock time setting register

This register enables the RTCAnHOUR, RTCAnMIN, and RTCAnSEC buffer registers to be read or written simultaneously.

Access This register can be read or written in 32-bit units.

Address <RTCAn_base> + 30_H

Initial Value 0012 0000_H when RTCAnCTL0.RTCAnAMPM = 0,
0000 0000_H when RTCAnCTL0.RTCAnAMPM = 1.

This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	RTCAnHOUR[7:0]							
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnMIN[7:0]								RTCAnSEC[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-40 RTCAnTIME register contents

Bit position	Bit name	Function
23 to 16	RTCAnHOUR [7:0]	Hours in BCD. See *Table 18-29 “12 and 24 hour format” on page 1311* for details.
15 to 8	RTCAnMIN [7:0]	Minutes in BCD.
7 to 0	RTCAnSEC [7:0]	Seconds in BCD.

Note Perform RTCAnTIME read/write as described in

- *18.5.1 “Initial setting of the RTCA” on page 1328*,
- *18.5.2 “Updating clock counters” on page 1330*, and
- *18.5.3 “Reading clock counters” on page 1331*.

(3) RTCAnCALC – RTCA calendar reading register

This register enables the RTCAnYEARC, RTCAnMONC, RTCAnDAYC, and RTCAnWEEKC counters to be read simultaneously.

<R> **Access** This register can be read in 32-bit units.

Address <RTCAn_base> + $6C_H$

Initial Value $0001\ 0100_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCTAnYEARC[7:0]								RTCAnMONC[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnDAYC[7:0]								RTCAnWEEKC[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 18-41 RTCAnCALC register contents

Bit position	Bit name	Function
31 to 24	RTCAnYEARC [7:0]	Year in BCD.
23 to 16	RTCAnMONC [7:0]	Month of the year in BCD.
15 to 8	RTCAnDAYC [7:0]	Day of the month in BCD.
7 to 0	RTCAnWEEKC [7:0]	Day of the week in BCD.

Notes

1. Perform RTCAnCALC read according to the flow described in *18.5.3 “Reading clock counters” on page 1331*.
2. A start value can be assigned to this register by writing to the clock time setting register RTCAnCAL. See
   - *18.5.1 “Initial setting of the RTCA” on page 1328* and
   - *18.5.2 “Updating clock counters” on page 1330*.

(4) RTCAnCAL – RTCA calendar setting register

This register enables the RTCAnYEAR, RTCAnMONTH, RTCAnDAY, and RTCAnWEEK buffer registers to be read or written simultaneously.

Access This register can be read or written in 32-bit units.

Address <RTCAn_base> + 34_H

Initial Value $0001\ 0100_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCTAnYEAR[7:0]								RTCAnMONTH[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnDAY[7:0]								RTCAnWEEK[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-42 RTCAnCAL register contents

Bit position	Bit name	Function
31 to 24	RTCAnYEAR [7:0]	Year in BCD.
23 to 16	RTCAnMONTH [7:0]	Month of the year in BCD.
15 to 8	RTCAnDAY [7:0]	Day of the month in BCD.
7 to 0	RTCAnWEEK [7:0]	Day of the week in BCD.

Note Perform RTCAnCAL read/write as described in

- *18.5.1 “Initial setting of the RTCA” on page 1328*,
- *18.5.2 “Updating clock counters” on page 1330*, and
- *18.5.3 “Reading clock counters” on page 1331*.

18.4.6 RTCA alarm setting register details

(1) RTCAnALM – RTCA alarm minute setting register

This register specifies the minute of the alarm interrupt.

For details and example settings, see *18.3.4 "Alarm interrupt function" on page 1291*.

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + 40_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	RTCAnALM[6:0]						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-43 RTCAnALM register contents

Bit position	Bit name	Function
6 to 0	RTCAnALM [6:0]	Minute of the alarm interrupt in BCD.

Notes
1. If decimal values outside the range of 00 to 59 in BCD are set, no alarm interrupt request will be generated.
2. When the setting of RTCAnALM is changed during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1), a glitch may be output to RTCATINTAL. Implement appropriate interrupt mask processing procedures.

(2) RTCAnALH – RTCA alarm hour setting register

This register specifies the hour of the alarm interrupt.

For details and example settings, see *18.3.4 "Alarm interrupt function" on page 1291*.

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + 44_H

Initial Value 12_H when RTCAnCTL0.RTCAnAMPM = 0,
00_H when RTCAnCTL0.RTCAnAMPM = 1.

This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	RTCAnALH[5:0]					
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-44 RTCAnALH register contents

Bit position	Bit name	Function
5 to 0	RTCAnALH [5:0]	Hour of the alarm interrupt in BCD.

Notes

1. If decimal values outside the following range are set, no alarm interrupt request will be generated:
   - 12 hour format (RTCAnCTL0.RTCAnAMPM = 0):
     01 to 12 or 21 to 32
   - 24 hour format (RTCAnCTL0.RTCAnAMPM = 1):
     00 to 23
2. When the setting of RTCAnALH is changed during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1), a glitch may be output to RTCATINTAL. Implement appropriate interrupt mask processing procedures.

(3) RTCAnALW – RTCA alarm day of the week setting register

This register specifies the day(s) of the week of the alarm interrupt.

Access This register can be read or written in 8-bit units.

Address <RTCAn_base> + 48_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	RTCAnALW[6:0]						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-45 RTCAnALW register contents

Bit position	Bit name	Function
6 to 0	RTCAnALW [6:0]	Specifies day of the week m (m = 0 to 6) as a day, where an alarm interrupt request is generated: 0: No alarm interrupt request is generated on day m 1: Alam interrupt request is generated on day m at the time set using RTCAnALM and RTCAnALH The bits of this register correspond to the count value of the day of the week counter (RTCAnWEEKC).

Note When the setting of RTCAnALW is changed during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1), a glitch may be output to RTCATINTAL. Implement appropriate interrupt mask processing procedures.

Example If Sunday is RTCAnWEEK = 0, Monday is RTCAnWEEK = 1, Tuesday is RTCAnWEEK = 2, …, Saturday is RTCAnWEEK = 6:

- To set the alarm to Sunday, set RTCAnALW = 0000 0001.
- To set the alarm to Monday and Wednesday, set RTCAnALW = 0000 1010.
- To set the alarm to Tuesday, Thursday, and Saturday, set RTCAnALW = 0101 0100

For more examples, see *18.3.4 “Alarm interrupt function” on page 1291*.

18.5 Procedures for Setup, Writing and Reading

The following subsections provide flow charts that illustrate the procedures for RTCA setup and for reading and writing the RTCA clock counters.

18.5.1 Initial setting of the RTCA

The RTCA must be stopped before setting initial counter values.

(1) RTCA stop procedure

Stop the RTCA according to the following flow.

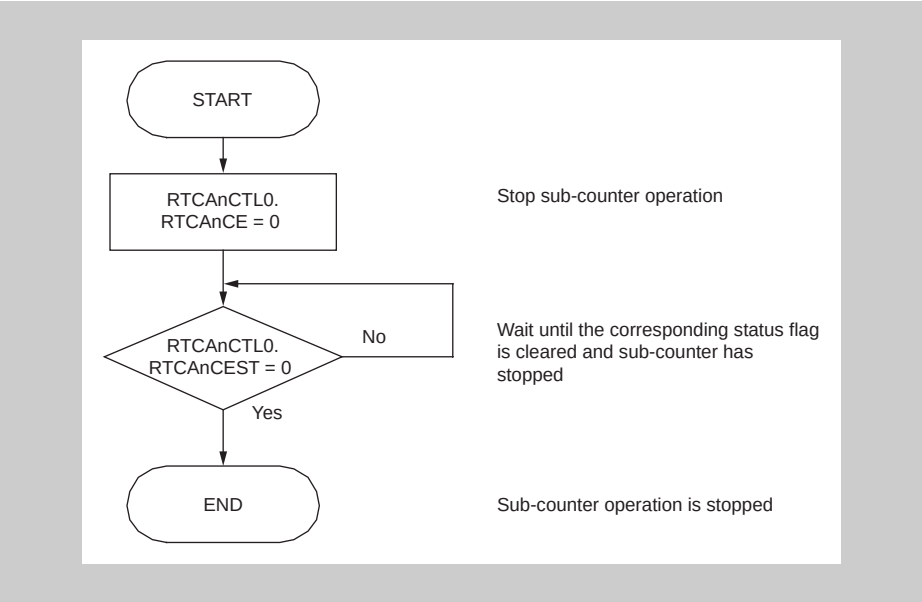


Figure 18-4 RTCA stop procedure

(2) RTCA initialization procedure

Perform the initial setting of the RTCA according to the following flow:

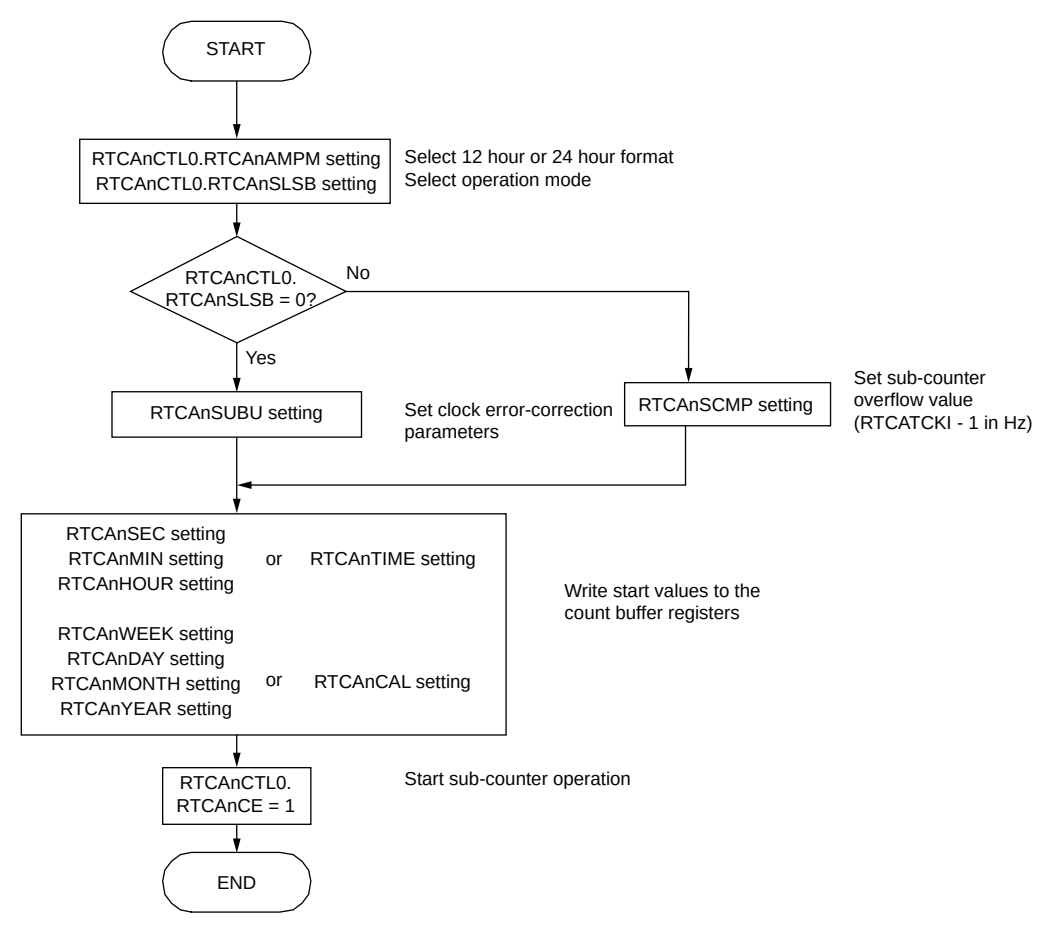


Figure 18-5 RTCA setup procedure

n-chip peripheral I/O are mapped to the on-chip peripheral I/O area.

Caution The register settings are becoming effective in synchronization with the RTCACKI clock. Consequently rewriting a register within a single RTCACKI period overwrites the previous register setting, which is lost in such case.

Thus do not rewrite a register within the flow of the above procedure.

18.5.2 Updating clock counters

The clock counters RTCAnSECC to RTCAnYEARC can be stopped and updated while the sub-counter is running.

The update procedure is illustrated in the following flow.

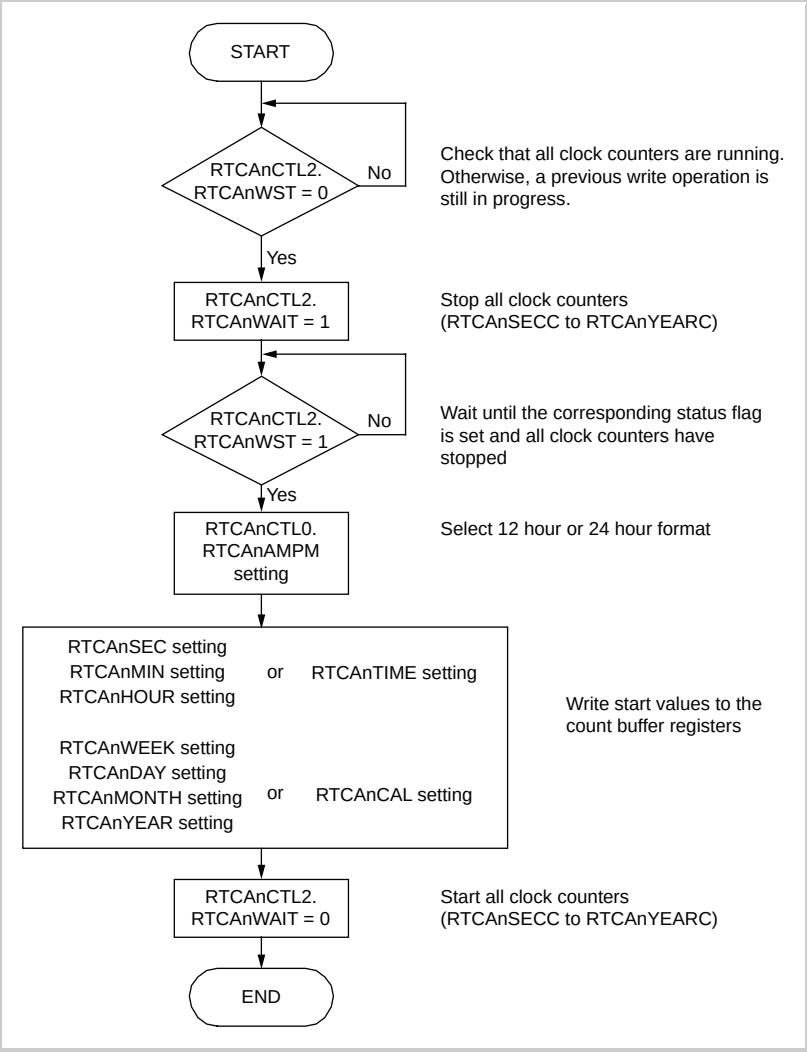


Figure 18-6 Updating clock counter values

Caution The register settings are becoming effective in synchronization with the RTCACKI clock. Consequently rewriting a register within a single RTCACKI period overwrites the previous register setting, which is lost in such case.

Thus do not rewrite a register within the flow of the above procedure.

Note The update procedure must be completed within 1 second. Otherwise the Real-Time Clock will not count correctly any more:

- Only *one* sub-counter overflow can be held internally and increment the seconds counter after restarting the clock counters.

- If the sub-counter overflows more than once during clock counter stop, the overflow count cannot be held internally. Thus the seconds counter is incremented by one instead of by two when it is restarted.

18.5.3 Reading clock counters

There are two methods to read the clock counters while sub-counter operation is enabled:

- Reading count buffer registers
- Reading counter registers

The advantages and disadvantages of the two methods are summarized in the following table.

Table 18-46 Comparison of the two read methods

	Advantage	Disadvantage
Reading count buffer registers	It is unnecessary to read clock counters several times because the clock counters are read synchronously.	A program wait state occurs between setting RTCAnCTL2.RTCAnWAIT = 1 and completion of data transfer.
Reading count registers	Program wait state does not occur.	The clock counters must be read several times because they are read asynchronously to RTCATCKI.

(1) Procedure for reading count buffer registers

The following operations are necessary:

1. Stop all clock counters (RTCAnCTL2.RTCAnWAIT = 1)
   The value of the clock counters is transferred to the count buffer registers.
2. Read the count buffer registers

A program wait state occurs between setting RTCAnCTL2.RTCAnWAIT = 1 and completion of data transfer.

The maximum delay is three PCLK periods plus two RTCATCKI periods. For example, if the RTCA operates with PCLK = 80 MHz and RTCATCKI = 32.678 kHz, the delay is about 61 µsec.

For a timing example, see *18.6.2 “Timing of RTCA while counter is enabled” on page 1338*.

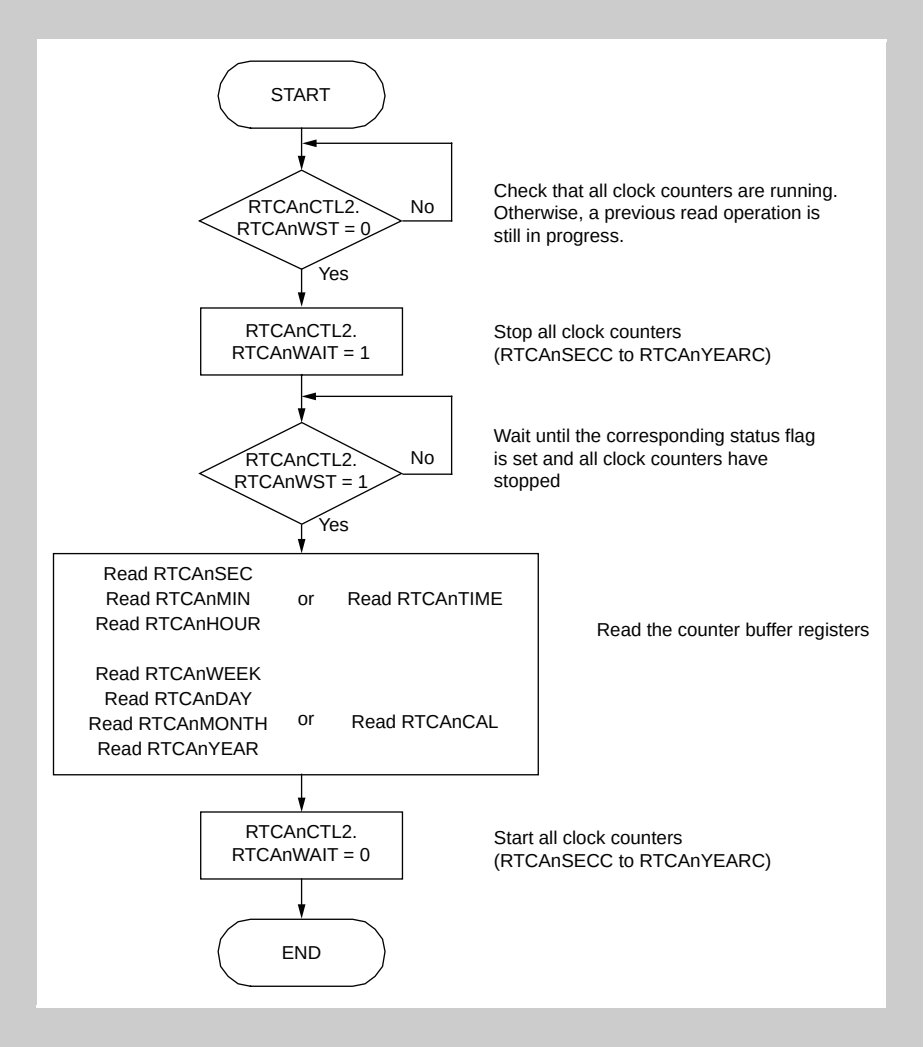


Figure 18-7 Reading clock count buffer registers

Note The update procedure must be completed within 1 second. Otherwise the Real-Time Clock will not count correctly any more:

- Only *one* sub-counter overflow can be held internally and increment the seconds counter after restarting the clock counters.
- If the sub-counter overflows more than once during clock counter stop, the overflow count cannot be held internally. Thus the seconds counter is incremented by one instead of by two when it is restarted.

(2) Procedure for reading counter registers directly

To ensure that the sub-counter did not overflow while reading the counters, the seconds counter RTCAnSECC must be read twice in the beginning and at the end of the procedure. The first read value is compared with the second read value.

- First read value = second read value:

  No overflow of sub-counter occurred during counter read operation.

- First read value <> second read value:

  Overflow of the sub-counter occurred during counter read operation. The counters must be read again to get the current counter values.

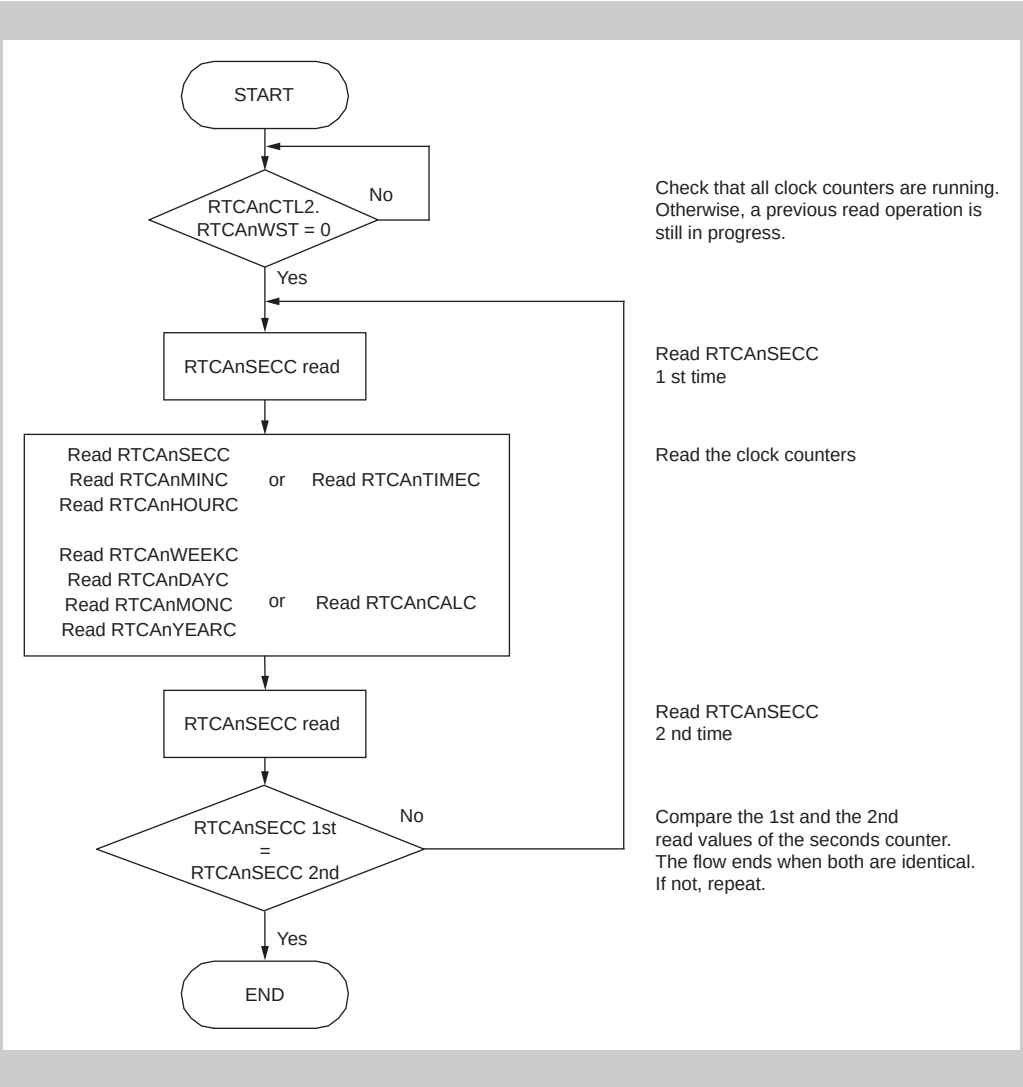


Figure 18-8 Reading clock counter registers

Note The procedure must be completed within 1 second.

18.5.4 Reading RTCAnSRBU

RTCAnSRBU is the read buffer register for the sub-counter.

When the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), read RTCAnSRBU according to the following flow.

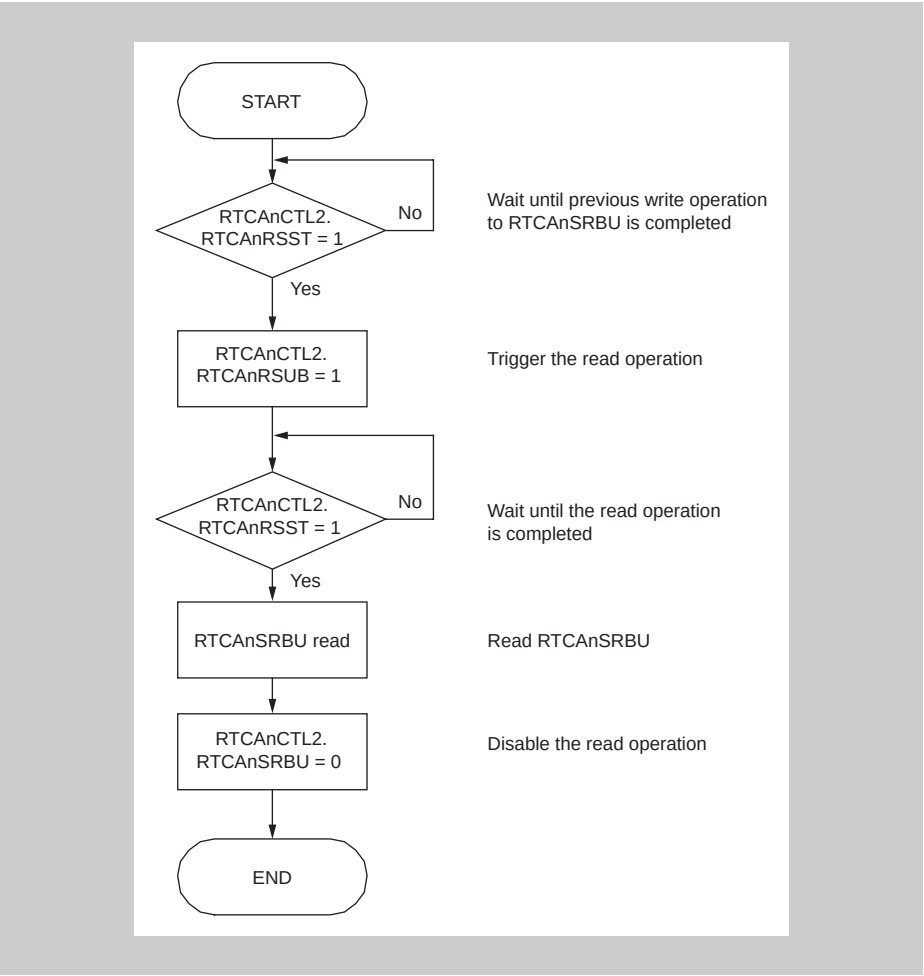


Figure 18-9 Reading the RTCAnSRBU register

18.5.5 Writing to RTCAnSUBU

RTCAnSUBU is the clock error correction register for the sub-counter.

When the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), write to RTCAnSUBU according to the flow described below.

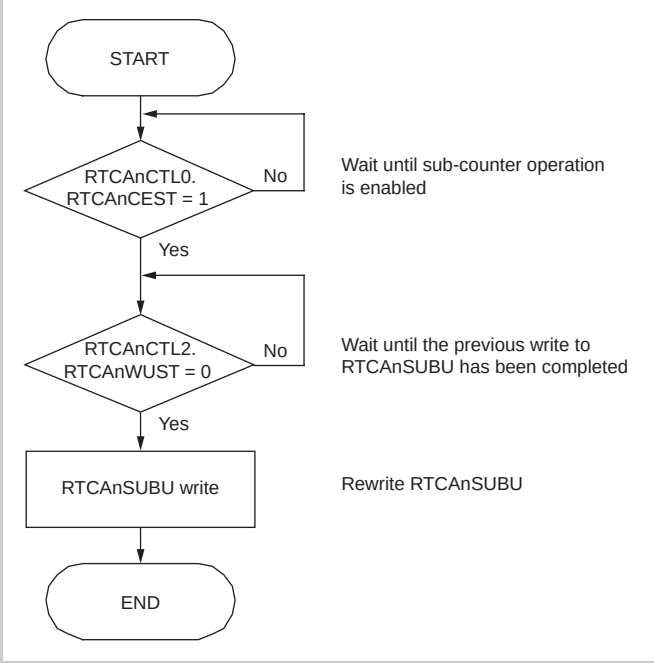


Figure 18-10 Writing to the RTCAnSUBU register

Note While the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), the status flag RTCAnCTL2.RTCAnWUST is set when RTCAnSUBU is written to. It is cleared when the write operation to RTCAnSUBU is completed. This is synchronous with the next RTCAnSUBC overflow.

RTCAnCTL2.RTCAnWUST can be set for up to 1 second.

18.5.6 Writing to RTCAnSCMP

RTCAnSCMP is the sub-counter compare register.

When the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), write to RTCAnSCMP according to the flow described below.

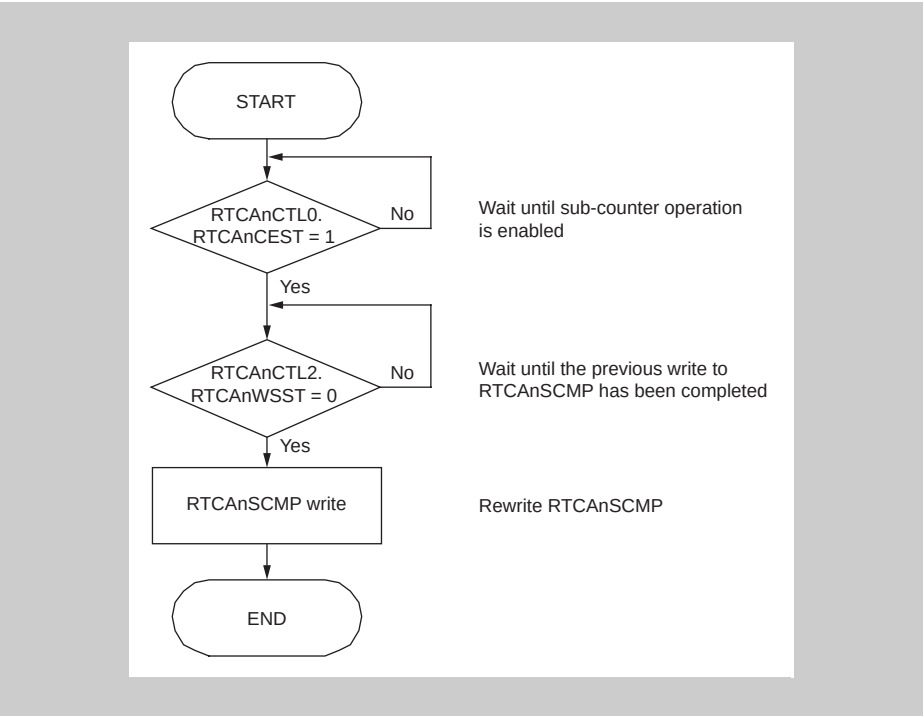


Figure 18-11 Writing the RTCAnSCMP register

Note While the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), the status flag RTCAnCTL2.RTCAnWSST is set when RTCAnSCMP is written to. It is cleared when the write operation to RTCAnSCMP is completed. This is synchronous with the next RTCAnSUBC overflow.

RTCAnCTL2.RTCAnWSST can be set for up to 1 second.

18.6 Timing Diagrams

18.6.1 Timing of RTCA counter start

The following diagram illustrates the counter start after setting the time in the buffer registers.

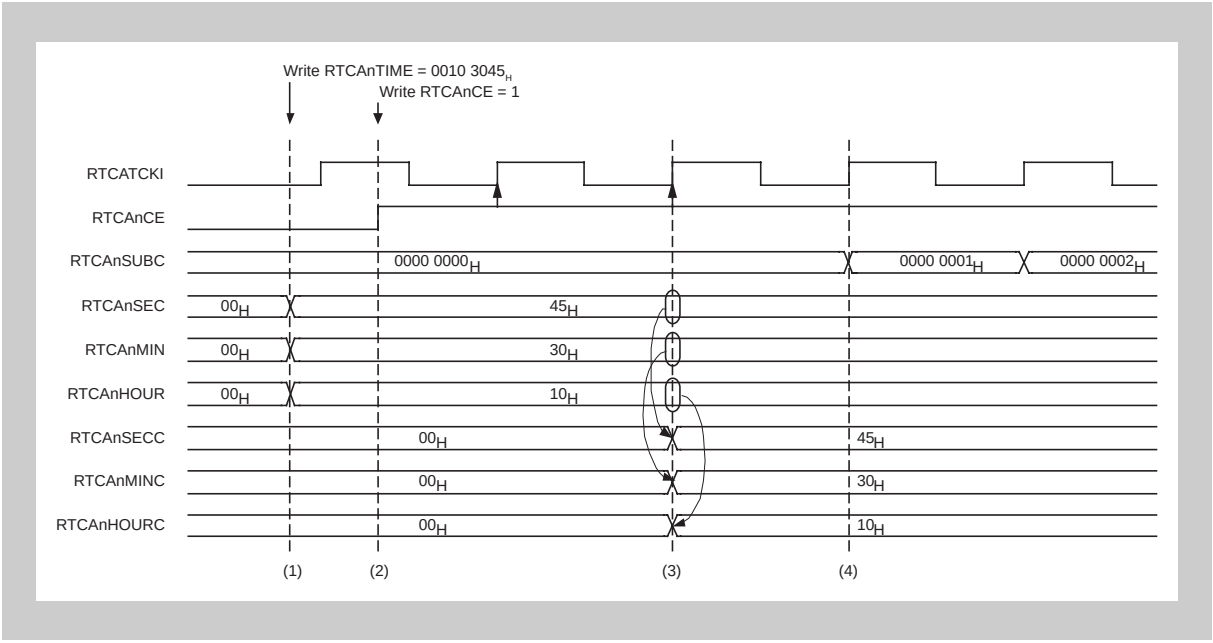


Figure 18-12 RTCA counter start timing

The timing diagram above shows the following:

1. The initial value of the time counters is set to 10:30:45 by writing RTCAnTIME= 0010 3045_H.
   Count buffer registers RTCAnSEC, RTCAnMIN, and RTCAnHOUR are automatically written.
2. Sub-counter operation is started by setting RTCAnCTL0.RTCAnCE = 1.
3. When the second rising edge of RTCATCKI occurs, both internal synchronization signals are set and the buffer register values are loaded to the count registers.
4. When the next rising edge of RTCATCKI occurs, count up of the sub-counter starts.

18.6.2 Timing of RTCA while counter is enabled

The following diagram illustrates the counter continuation after setting the time in the buffer registers.

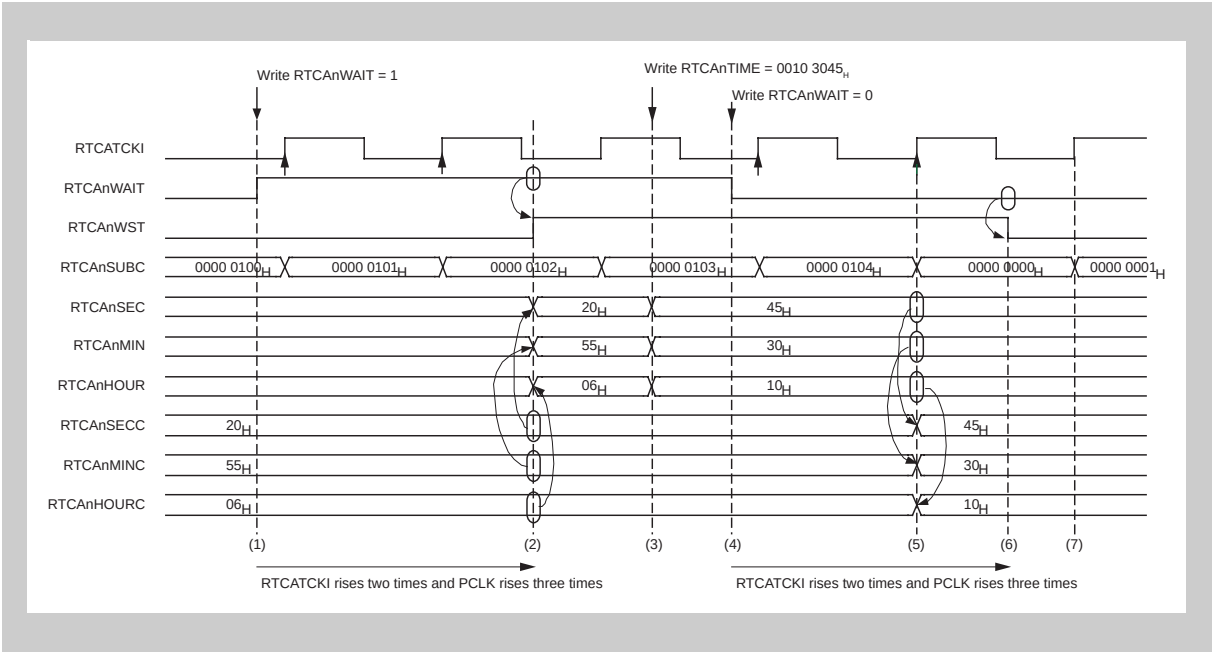


Figure 18-13 RTCA counter continuation timing

The timing diagram above shows the following:

1. Trigger the clock counters stop (RTCAnCTL2.RTCAnWAIT = 1).
2. It takes at least one RTCATCKI period to stop the counters. Stop of the counters is reflected by RTCAnCTL2.RTCAnWST = 1.

   The sub-counter continues counting.
3. The initial value of the time counters is set to 10:30:45 by writing RTCAnTIME= 0010 3045_H.

   Count buffer registers RTCAnSEC, RTCAnMIN, and RTCAnHOUR are automatically written.
4. Trigger the clock counters restart (RTCAnCTL2.RTCAnWAIT = 0).
5. After at least one RTCATCKI period RTCAnSECC is written and RTCAnSUBC is cleared.
6. The clock counters are ready for counter re-start (RTCAnCTL2.RTCAnWST = 0).
7. Clock counter operation is resumed.

18.6.3 Timing of sub-counter buffer read while counter is enabled

The following diagram illustrates the timing when reading the sub-counter read buffer RTCAnSRBU.

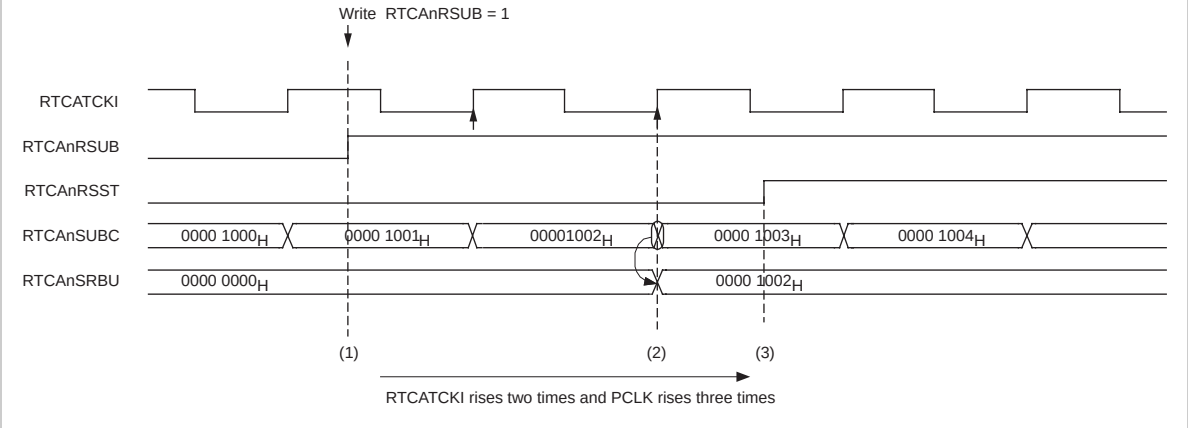


Figure 18-14 Timing when reading the sub-counter buffer register value

The timing diagram above shows the following:

1. Transfer of RTCAnSRBU value to the sub-counter is triggered (RTCAnCTL2.RTCAnRSUB = 1).
2. When the second rising edge of RTCATCKI occurs, the value of RTCAnSUBC is loaded to RTCAnSRBU.
3. Completion of the transfer is indicated by RTCAnCTL2.RTCAnRSST = 1. RTCAnSRBU can now be read.

Chapter 19 Encoder Timer (ENCA)

This chapter describes the encoder timer (ENCA).

The first section describes the properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names.
The subsequent sections describe the features that apply to all implementation.

19.1 V850E2/Sx4-H ENCA Features

Instances

This microcontroller has the following number of instances of ENCA:

Table 19-1 Instances of ENCA

ENCA	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of instances	0	0	2
Name	–	–	ENCA0, ENCA1

Instances index n

Throughout this chapter, the individual instances of ENCA is identified by the index "n" (n = 0, 1), for example, ENCAnCTL for the ENCAn control register.

Register addresses

All ENCAn register addresses are given as addresses offset from the individual base address <ENCAn_base>.
The base address <ENCAn_base> of each ENCAn is listed in the following table:

Table 19-2 Register base addresses <ENCAn_base>

ENCAn	<ENCAn_base> address
ENCA0	FF81 9000_H
ENCA1	FF81 $A000_H$

Clock supply

The following clocks are supplied to ENCA:

Table 19-3 ENCAn clock supply

ENCAn	Clock	Connected to:
ENCA0	PCLK	Clock generator CKSCLK_006
ENCA1	PCLK	Clock generator CKSCLK_006

Interrupts

ENCA can generate the following interrupt requests:

Table 19-4 ENCAn interrupt requests

ENCAn signals	Function	Connected to:
ENCA0:		
ENCATIOV	ENCA0 overflow interrupt	Interrupt controller INTENCA0IOV
ENCATIUD	ENCA0 underflow interrupt	Interrupt controller INTENCA0IUD
ENCATINT0	ENCA0 compare match 0 or capture 0 interrupt	Interrupt controller INTENCA0I0
ENCATINT1	ENCA0 compare match 1 or capture 1 interrupt	Interrupt controller INTENCA0I1
ENCATIEC	ENCA0 clear interrupt by encoder input (phase Z)	Interrupt controller INTENCA0IEC
ENCA1:		
ENCATIOV	ENCA1 overflow interrupt	Interrupt controller INTENCA1OV
ENCATIUD	ENCA1 underflow interrupt	not connected
ENCATINT0	ENCA1 compare match 0 or capture 0 interrupt	Interrupt controller INTENCA1CC0
ENCATINT1	ENCA1 compare match 1 or capture 1 interrupt	Interrupt controller INTENCA1CC1
ENCATIEC	ENCA1 clear interrupt by encoder input (phase Z)	Interrupt controller INTENCA1EC

ENCA hardware reset

ENCA and its registers are initialized by the following reset signal:

Table 19-5 ENCAn reset signal

ENCAn	Reset signal
ENCAn	System reset SYSRES

I/O signals The I/O signals of ENCA are listed in the following table:

Table 19-6 ENCAn I/O signals

ENCAn signal	Function	Connected to:
ENCA0:		
ENCATTIN0	ENCA0 capture trigger input 0	Port ENC0TTIN0
ENCATTIN1	ENCA0 capture trigger input 1	Port ENC0TTIN1
ENCATAIN	ENCA0 encoder input (phase A)	Port ENC0TAIN
ENCATBIN	ENCA0 encoder input (phase B)	Port ENC0TBIN
ENCATZIN	ENCA0 encoder input (phase Z)	Port ENC0TZIN
ENCATSST	ENCA0 simultaneous start trigger input	Fixed to low level
ENCATSTT	ENCA0 simultaneous stop trigger input	Fixed to low level
ENCA1:		
ENCATTIN0	ENCA1 capture trigger input 0	Port ENC1TTIN0
ENCATTIN1	ENCA1 capture trigger input 1	Port ENC1TTIN1
ENCATAIN	ENCA1 encoder input (phase A)	Port ENC1TAIN
ENCATBIN	ENCA1 encoder input (phase B)	Port ENC1TBIN
ENCATZIN	ENCA1 encoder input (phase Z)	Port ENC1TZIN
ENCATSST	ENCA1 simultaneous start trigger input	Fixed to low level
ENCATSTT	ENCA1 simultaneous stop trigger input	Fixed to low level

19.2 Functional Overview

Features summary

- Generation of the counter control signal from the encoder input signal, and performs count operation in synchronization with PCLK.
- Capture function for capturing the counter value with an external trigger signal
- Compare function for compare match judgment with the counter value
- Two capture compare registers that can be set separately for capture operation and for compare operation
- Interrupt mask function for masking the interrupt signal output as a result of compare match judgment during compare operation
- Function for loading the value of the capture compare register to the counter upon underflow occurrence
- Encoder input signal can be applied to the timer counter clear condition
- Edge or level for clearing the encoder input signal of the timer counter clear condition can be selected
- Detection of counter overflow and underflow and output error flags and occurrence interrupts
- Five interrupts: 2 capture compare interrupts, 1 counter clear interrupt, 1 overflow interrupt, and 1 underflow interrupt.

19.2.1 Block diagram

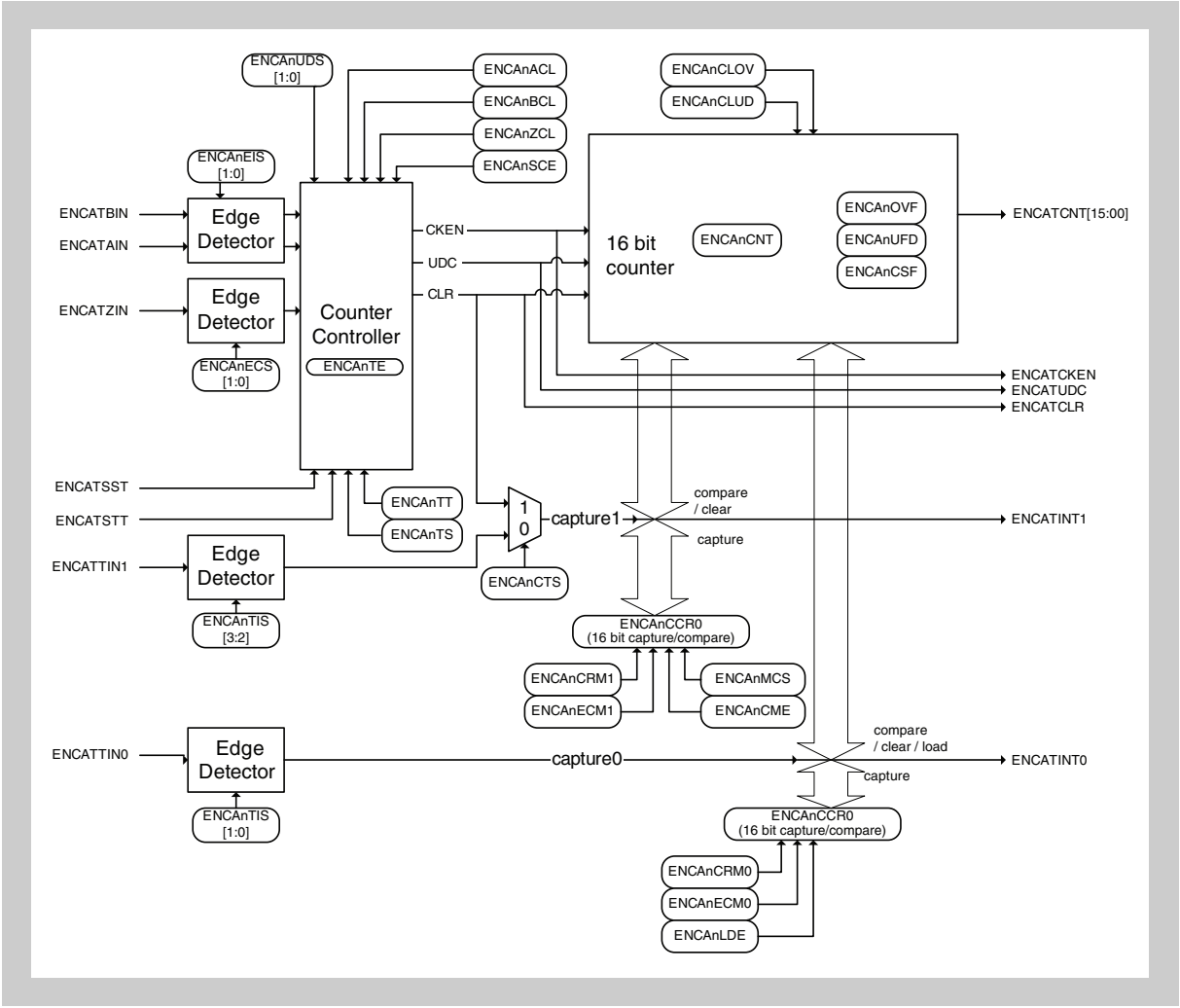


Figure 19-1 Encoder timer block diagram

19.2.2 Preliminary knowledge for understanding basic specifications

As this macro has an encoder counter function, the rotary encoder is explained below.

Rotary encoder

The rotary encoder outputs a pulse according to the amount of displacement of the axis of rotation. The output pulse is output as a waveform that has two phase differences, from the fact that the light passing through two slits (phase A, phase B) is transmitted or blocked, when the disc, on which an optical pattern is written, rotates, along with the displacement of the axis of rotation.

The rotation can be detected by counting the output pulses with a counter.

Moreover, there is a pulse called phase Z, which indicates the origin within 1 rotation of the encoder.

Errors can be corrected by clearing the counter to 0000_H by using the phase Z pulse.

Position measurement and speed measurement applications can be achieved by using the rotation measured with the timer counter.

19.3 ENCA Control Registers

The ENCAn is controlled and operated by the following registers:

Table 19-7 ENCAn register overview

Register name	Symbol	Address
ENCA capture compare register 0	ENCAnCCR0	<ENCAn_base>
ENCA capture compare register 1	ENCAnCCR1	<ENCAn_base> + 04_H
ENCA counter register	ENCAnCNT	<ENCAn_base> + 08_H
ENCA status flag register	ENCAnFLG	<ENCAn_base> + $0C_H$
ENCA status flag clear register	ENCAnFGC	<ENCAn_base> + 10_H
ENCA timer enable status register	ENCAnTE	<ENCAn_base> + 14_H
ENCA timer start trigger register	ENCAnTS	<ENCAn_base> + 18_H
ENCA timer stop trigger register	ENCAnTT	<ENCAn_base> + $1C_H$
ENCA I/O control register 0	ENCAnIOC0	<ENCAn_base> + 20_H
ENCA control register	ENCAnCTL	<ENCAn_base> + 40_H
ENCA I/O control register 1	ENCAnIOC1	<ENCAn_base> + 44_H

<ENCAn_base> The base addresses <ENCAn_base> of the ENCAn is defined in the first section of this chapter under the key word “Register addresses”.

(1) ENCAnCTL – ENCA control register

This register is used to configure various operation settings of the Encoder Timer.

Access This register can be read/written in 16-bit units.
Writing to this register during operation is prohibited.

Address <ENCAn_base> + 40_H

Initial value Reset input initializes this register to 0000_H.

15	14	13	12	11	10	9	8
ENCAn CME	ENCAn MCS	0	0	0	0	ENCAn CRM1	ENCAn CRM0
R/W	R/W	R	R	R	R	R/W	R/W

7	6	5	4	3	2	1	0
ENCAn CTS	0	0	ENCAn LDE	ENCAn ECM1	ENCAn ECM0	ENCAn UDS[1:0]	
R/W	R	R	R/W	R/W	R/W	R/W	R/W

Table 19-8 ENCAnCTL register contents (1/2)

Bit	Name	Function
15	ENCAnCME	Encoder clear mask enable bit This bit is used to enable/disable masking of compare match interrupt detection when the compare function is used. 0: Disables the compare match interrupt (ENCATINT1) mask function for the ENCAnCCR1 register 1: Enables the compare match interrupt (ENCATINT1) mask function for the ENCAnCCR1 register. This bit is valid only when ENCAnCRM1 = 0. When this bit is set to 1, setting ENCAnECM1 to 1 is prohibited.
14	ENCAnMCS	Encoder mask clear select bit This bit is used to select the trigger for cancelling masking of compare match interrupt detection when the compare function is used. This bit is valid only when ENCAnCRM1 = 0. 0 Masking of compare match interrupt detection is cancelled when the ENCAnCCR1 register is written. 1: Masking of compare match interrupt detection is cancelled when one of the following three operations is performed. -Timer counter clear operation accompanying phase Z -Timer counter clear operation upon compare match between ENCAnCNT and ENCAnCCR0 when ENCAnECM0 = 1 -Loading from ENCAnCCR0 to timer counter upon underflow detection when ENCAnLDE = 1
9	ENCAnCRM1	ENCACCR1 register mode bit 0: ENCAnCCR1 used as compare register. 1: ENCAnCCR1 used as capture register.
8	ENCAnCRM0	ENCACCR0 register mode bit 0: ENCAnCCR0 used as compare register. 1: ENCAnCCR0 used as capture register.

Table 19-8 ENCAnCTL register contents (2/2)

Bit	Name	Function
7	ENCAnCTS	ENCAnCCR1 capture trigger select bit This is a trigger selection bit register for the capture operation to the ENCAnCCR1 register. This bit selects the capture trigger to the ENCAnCCR1 register. This bit is valid only when ENCAnCRM1 = 1. 0: Uses ENCATTIN1 of capture trigger 1 signal as the trigger for capturing to the ENCAnCCR1 register. 1: Uses ENCATZIN of the phase Z encoder input signal as the trigger for capturing to the ENCAnCCR1 register.
4	ENCAnLDE	ENCAn counter load enable bit This register is used to enable/disable setting value loading to the counter upon underflow occurrence. This bit is valid only when ENCAnCRM0 = 0. When ENCAnCRM0 = 1, loading of the ENCAnCCR0 register setting value to the counter upon occurrence of an underflow is not performed, regardless of the value of this bit. 0: Disable loading of ENCAnCCR0 register setting value to counter upon occurrence of a counter underflow. 1: Enable loading of ENCAnCCR0 register setting value to counter upon occurrence of a counter underflow.
3	ENCAnECM1	Encoder clear mode bit 1 This register is used to set the counter clear operation upon match between the counter value and ENCAnCCR1 setting value. This bit is valid only when ENCAnCRM1 = 0. 0: Does not clear the counter to 0000_H upon match of timer counter value and ENCAnCCR1 setting value. 1: Clears the counter to 0000_H upon match of timer counter value and ENCAnCCR1 setting value if the next count is a up-count.
2	ENCAnECM0	Encoder clear mode bit 0 This register is used to set the counter clear operation upon match between the counter value and ENCAnCCR0 setting value. This bit is valid only when ENCAnCRM0 = 0. 0: Does not clear the counter to 0000_H upon match of timer counter value and ENCAnCCR0 setting value. 1: Clears the counter to 0000_H upon match of timer counter value and ENCAnCCR0 setting value if the next count is a down-count.
1, 0	ENCAn UDS[1:0]	UPDOWN count selection bits 1 and 0 This is the counter up/down control register using ENCATAIN and ENCATBIN. 00: Upon detection of valid edge of ENCATAIN, - down-count when ENCATBIN = H, - up-count when ENCATBIN = L 01: Upon detection of valid edge of ENCATAIN, up-count, Upon detection of valid edge of ENCATBIN, down-count 10: At rising edge of ENCATAIN, down-count At falling edge of ENCATAIN, up-count However, count operation performed only when ENCATBIN = L. 11: Detection of both edges of ENCATAIN, ENCATBIN. Judgment of count operation combining both detected edge and level.

(2) ENCAnIOC0 – ENCA I/O control register 0

This register is used to select the input edge of capture triggers 0 and 1 (ENCAnTTIN0, ENCAnTTIN1).

Access This register can be read/written in 8-bit units.
Writing to this register during operation is prohibited.

Address <ENCAn_base> + 20_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	ENCAnTIS[3:2]		ENCAnTIS[1:0]	
R	R	R	R	R/W	R/W	R/W	R/W

Table 19-9 ENCAnIOC0 register contents

Bit	Name	Function
3, 2	ENCAn TIS[3:2]	Input edge selection bits for capture trigger 1. These bits are valid only when the ENCAnCTL register's ENCAnCRM1 = 1 and ENCAnCTS = 0. All other settings of ENCAnCRM1 and ENCAnCTS are invalid. 00: No edge detection 01: Rising edge detection 10: Falling edge detection 11: Both edges detection
1, 0	ENCAn TIS[1:0]	Input edge selection bits for capture trigger 0. These bits are valid only when ENCAnCTL.ENCAnCRM0 = 1 00: No edge detection 01: Rising edge detection 10: Falling edge detection 11: Both edges detection

(3) ENCAnIOC1 – ENCA I/O control register 1

This register is used to perform the clear condition setting and edge selection upon encoder input.

Access This register can be read/written in 8-bit units.
Writing to this register during operation is prohibited.

Address <ENCAn_base> + 44_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
ENCAn SCE	ENCAn ZCL	ENCAn BCL	ENCAn ACL	ENCAn ECS[1:0]		ENCAn EIS[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-10 ENCAnIOC1 register contents (1/2)

Bit	Name	Function
7	ENCAnSCE	Encoder special-clear enable bit This is an encoder special clear enable bit. When cleared to 0, the counter is cleared upon phase Z edge detection. When set to 1, the counter is cleared using the level of ENCATZIN, ENCATAIN, and ENCATBIN (special clear). When setting this bit to 1, set ENCAnUDS1 and ENCAnUDS0 to {1, 0} or {1, 1}. The operation is not guaranteed if this bit is set to 1 with ENCAnUDS1 and ENCAnUDS0 set to {0, 0} or {0, 1}. 0: Clears the counter upon detection of ENCATZIN valid edge (set with ENCAnECS1 and ENCAnECS0). 1: Clears the counter upon detection of input level condition of ENCATZIN, ENCATBIN and ENCATAIN (set with ENCAnZCL bit, ENCAnBCL bit, and ENCAnACL bit).
6	ENCAnZCL	Input-Z clear condition selection bit This bit is used to set the condition for clearing the phase Z encoder input (ENCAnZIN) when using the encoder special clear function. This bit is valid only when ENCAnSCE = 1; it is invalid when ENCAnSCE = 0. 0: Clear condition: Low level 1: Clear condition: High level
5	ENCAnBCL	Input-B clear condition selection bit This bit is used to set the condition for clearing the phase B encoder input (ENCAnBIN) when using the encoder special clear function. This bit is valid only when ENCAnSCE = 1; it is invalid when ENCAnSCE = 0. 0: Clear condition: Low level 1: Clear condition: High level

Table 19-10 ENCAnIOC1 register contents (2/2)

Bit	Name	Function
4	ENCAnACL	Input-A clear condition selection bit This bit is used to set the condition for clearing the phase A encoder input (ENCAnAIN) when using the encoder special clear function. This bit is valid only when ENCAnSCE = 1; it is invalid when ENCAnSCE = 0. 0: Clear condition: Low level 1: Clear condition: High level
3, 2	ENCAn ECS[1:0]	Encoder clear input edge selection bits 1 and 0 These are the encoder clear input edge selection bits (phase Z). These bits are valid only when ENCAnSCE = 0; they are invalid when ENCAnSCE = 1. 00: No edge detection 01: Rising edge detection 10: Falling edge detection 11: Both edges detection
1, 0	ENCAn EIS[1:0]	Encoder input selection bits 1 and 0 These are the encoder input edge selection bits (phase A, phase B). These bits are valid when ENCAnUDS1 and ENCAnUDS0 = {0, 0} or {0, 1}, and are invalid when ENCAnUDS1 and ENCAnUDS0 = {1, 0} or {1, 1}. 00: No edge detection 01: Rising edge detection 10: Falling edge detection 11: Both edges detection

(4) ENCAnFLG – ENCA status flag register

This register is holds the status flags of the timer counter of the ENCAn.

Access This register can be read in 8-bit units.
Writing to this register during operation is prohibited.

Address <ENCAn_base> + $0C_H$

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
					ENCAnCSF	ENCAnUDF	ENCAnOVF
R	R	R	R	R	R	R	R

Table 19-11 ENCAnFLG register contents

Bit	Name	Function
2	ENCAnCSF	Counter status flag This bit reflects the current timer counter operation. This bit is cleared at the start of count operation. 0: Timer counter in up-count status 1: Timer counter in down-count status
1	ENCAnUDF	Underflow flag This bit reflects the occurrence of an underflow during the timer counter operation. This bit is cleared at the start of count operation. 0: This flag is cleared upon any of the following events: - 1 is written to ENCAnFGC.ENCAnCLUD - ENCAnTS is set while ENCAnTE = 0 - The ENCASTT signal becomes high level 1: This flag is set to 1 upon occurrence of an underflow during the encoder timer count operation.
0	ENCAnOVF	Overflow flag This bit reflects the occurrence of an overflow during the timer counter operation. This bit is cleared at the start of count operation. 0: This flag is cleared upon any of the following events: - 1 is written to ENCAnFGC.ENCAnCLOV - ENCAnTS is set while ENCAnTE = 0 - The ENCASST signal becomes high level 1: This flag is set to 1 upon occurrence of an overflow during the encoder timer count operation.

(5) ENCAnFGC – ENCA status flag clear register

This register is used to clear the timer counter status flags of ENCAnFLG.

Access This register can be written in 8-bit units.
This register always returns 0 when read.

Address <ENCAn_base> + 10_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	ENCAnCLUD	ENCAnCLOV
W	W	W	W	W	W	W	W

Table 19-12 ENCAnFGC register contents

Bit	Name	Function
1	ENCAnCLUD	Underflow flag clear This bit clears the underflow flag. 0: Writing is ignored. 1: Clears ENCAnUDF of the ENCAnFL register (clears underflow detection).
0	ENCAnCLOV	Overflow flag clear This bit clears the overflow flag. 0: Writing is ignored. 1: Clears ENCAnOVF of the ENCAnFL register (clears overflow detection).

(6) ENCAnCCR0 – ENCA capture compare register 0

This register is a 16-bit capture compare register 0.

Access This register can be written in 16-bit units when used as a compare register (ENCAnCTL.ENCAnCRM0 = 0).
When used as a capture register (ENCAnCTL.ENCAnCRM0 = 1), writing to this register during operation is invalid.

Address <ENCAn_base> + 00_H

Initial value Reset input initializes this register to 0000_H.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENCAnCCR0[15:0]															
R/W															

Table 19-13 ENCAnCCR0 register contents

Bit	Name	Function
15 to 0	ENCAn CCR0[15:0]	Capture compare register 0 Upon occurrence of an underflow, the setting value of this register may be loaded to the counter according to the ENCAnCTL.ENCAnLDE setting. See the description of the ENCAnLDE bit in ENCA control register ENCAnCTL for details. • if ENCAnCTL.ENCAnCRM0 = 0: ENCAnCCR0 is compare register Set the value to be compared with the timer counter value. • if ENCAnCTL.ENCAnCRM0 = 1: ENCAnCCR0 is capture register The captured timer counter value is stored.

(7) ENCAnCCR1 – ENCA capture compare register 1

This register is a 16-bit capture compare register 1.

Access This register can be written in 16-bit units when used as a compare register (ENCAnCTL.ENCAnCRM1 = 0).
When used as a capture register (ENCAnCTL.ENCAnCRM1 = 1), writing to this register during operation is invalid.

Address <ENCAn_base> + 04_H

Initial value Reset input initializes this register to 0000_H.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENCAnCCR1[15:0]															
R/W															

Table 19-14 ENCAnCCR1 register contents

Bit	Name	Function
15 to 0	ENCAn CCR1[15:0]	Capture compare register 1 During capture operation, the trigger for capturing to this register differs according to the ENCAnCTL.ENCAnCTS setting. See the description of the ENCAnCTS bit in ENCA control register ENCAnCTL for details. • if ENCAnCTL.ENCAnCRM1 = 0: ENCAnCCR1 is compare register Set the value to be compared with the timer counter value. • if ENCAnCTL.ENCAnCRM1 = 1: ENCAnCCR1 is capture register The captured timer counter value is stored.

(8) ENCAnCNT – ENCA counter register

This register is the 16-bit timer counter register.

Access This register can be read/written in 16-bit units.
This register can be written only when the operation is stopped.

Address <ENCAn_base> + 08_H

Initial value Reset input initializes this register to 0000_H.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENCAnCNT[15:0]															
R/W															

Table 19-15 ENCAnCNT register contents

Bit	Name	Function
15 to 0	ENCAn CNT[15:0]	Counter register • ENCAnTE.ENCAnTE status: 0 (initial setting): Count stop An arbitrary value can be set to timer counter. • ENCAnTE.ENCAnTE status: 0 → 1 (operation start)): Count operation start Up/down count operation is started with the set arbitrary value. • ENCAnTE.ENCAnTE status: 1 (operating): Counting Up/down count operation is performed. • ENCAnTE.ENCAnTE status: 1 → 0 (stopped): Count stop The counter value immediately before the operation was stopped is held, and the count operation is stopped.

(9) ENCAnTE – ENCA timer enable status register

This register indicates the operating status of the ENCAn.

Access This register can be read in 8-bit units.

Address <ENCAn_base> + 14_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ENCAnTE
R	R	R	R	R	R	R	R

Table 19-16 ENCAnTE register contents

Bit	Name	Function
0	ENCAnTE	Timer status enable bit This is a status bit that indicates the operation enabled/stopped status of the ENCAn. This bit is cleared to 0 when 1 is written to ENCAnTT.ENCAnTT, or when a high level is input to the ENCATSTT signal. This bit is set to 1 when 1 is written to ENCAnTS.ENCAnTS, or when a high level is input to the ENCATSST signal. 0: Operation stopped status 1: Operation enabled status

(10) ENCAnTS – ENCA timer start trigger register

This register provides the trigger bit for setting the ENCAn to the operation enabled state.

Access This register can be read/written in 8-bit units.
This register always returns 0 when read. This register can be written only when ENCAnTE.ENCAnTE is 0.

Address <ENCAn_base> + 18_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ENCAnTS
R	R	R	R	R	R	R	W

Table 19-17 ENCAnTS register contents

Bit	Name	Function
0	ENCAnTS	Timer start trigger bit This is the trigger bit that sets the ENCAn to the operation enabled state. 0: Writing is ignored. 1: ENCAn is set to the operation enabled state by setting ENCAnTE.ENCAnTE = 1.

(11) ENCAnTT – ENCA timer stop trigger register

This register provides the trigger bit for setting the ENCAn to the operation stopped state.

Access This register can be read/written in 8-bit units.
This register always returns 0 when read.

Address <ENCAn_base> + $1C_H$

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ENCAnTT
R	R	R	R	R	R	R	W

Table 19-18 ENCAnTT register contents

Bit	Name	Function
0	ENCAnTT	Timer stop trigger bit This is the trigger bit that sets the ENCAn to the operation stopped state. 0: Writing is ignored. 1: Clears ENCAnTE.ENCAnTE to 0, to set the ENCAn to the count operation stopped state.

19.4 Functional Description

The ENCAn operates the timer counter with counter up/down control and clear control by encoder inputs (phase A, phase B, phase Z). The ENCAnCCR0 and ENCAnCCR1 registers can be used as dedicated compare registers or as dedicated capture registers.

19.4.1 Timer counter operation

The timer counter operations of the ENCAn are described below.

The figure below below shows the operation phases. See the corresponding section with the section number for detailed descriptions on each operation.

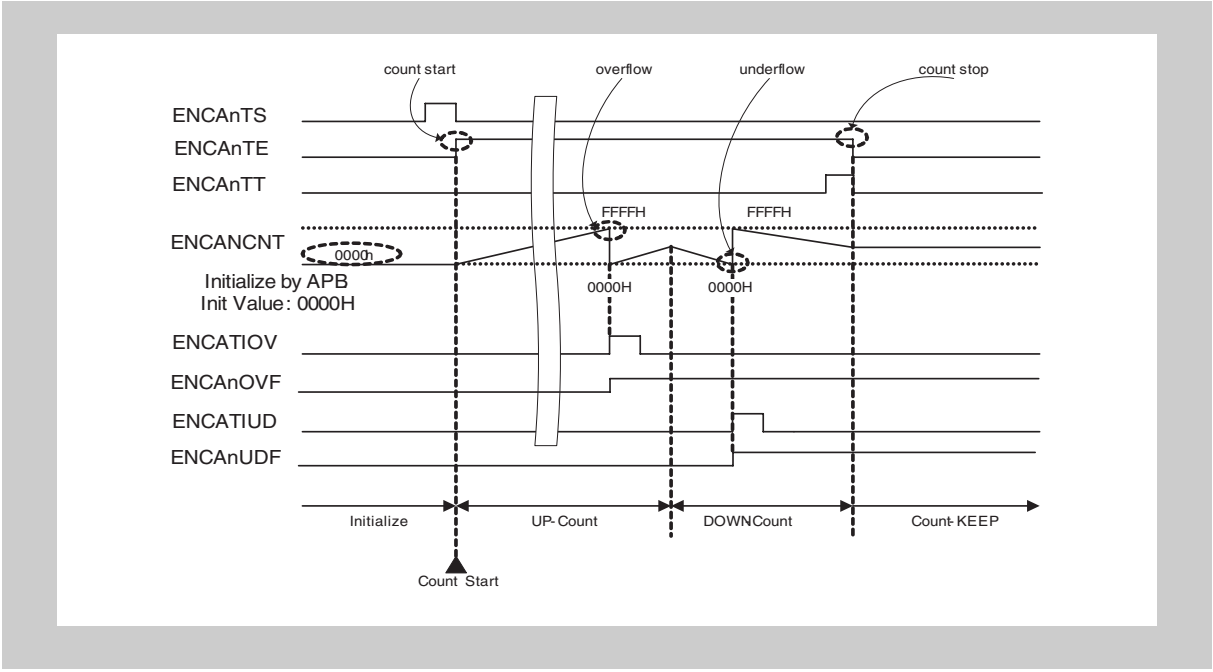


Figure 19-2 Timer counter initial value setting/start/stop

(1) Timer counter initial value setting

The initial value of the ENCAn counter register (ENCAnCNT) can be set in the counter operation stopped status (ENCAnTE = 0).

(2) Timer counter startup

By writing 1 to the timer start trigger bit (ENCAnTS), the timer status enable bit (ENCAnTE) is set to 1, the count operation is enabled, and counting operation is performed upon detection of the valid edge of the encoder input.

(3) Overflow operation

An overflow occurs when up-counting is performed when the counter value is $FFFF_H$. If the counter value changes from $FFFF_H$ to 0000_H, an overflow interrupt (ENCATIOV) is generated, and the overflow flag (ENCAnOVF) is set to 1. The overflow flag (ENCAnOVF) is cleared to 0 when 1 is set to the overflow flag clear bit (ENCAnCLOV).

(4) Underflow operation

An underflow occurs when down-counting is performed when the counter value is 0000_H. If the counter value changes from 0000_H to $FFFF_H$, an underflow interrupt (ENCATIUD) is generated, and the underflow flag (ENCAnUDF) is set to 1. The underflow flag (ENCAnUDF) is cleared to 0 when 1 is set to the underflow flag clear bit (ENCAnCLUD).

(5) Timer counter stop

By writing 1 to the timer stop trigger bit (ENCAnTT), the timer status enable bit (ENCAnTE) is cleared to 0, and the count operation is stopped. At this time, the timer counter is not reset to 0000_H and holds the value before count operation stop.

19.4.2 Up/down control of timer counter

Up/down control is performed by judging the phase of the encoder inputs (ENCTAIN, ENCTBIN) according to the settings of ENCAnUDS1 and ENCAnUDS0.

(1) When ENCAnUDS1 and ENCAnUDS0 = {0, 0}

ENCAnUDS1	ENCAnUDS0	Operation description		
		ENCATAIN input	ENCTBIN input	Count operation
0	0	Rising edge	High level	Down
		Falling edge		
		Both edges		
		Rising edge	Low level	Up
		Falling edge		
		Both edges		

The valid edge for ENCATAIN is specified by setting ENCAnEIS1 and ENCAnEIS0.

Count operation is performed when the valid edges of ENCATAIN and ENCATBIN overlap.

The following timing chart shows the count operation when ENCAnUDS1 and ENCAnUDS0 = {0, 0}.

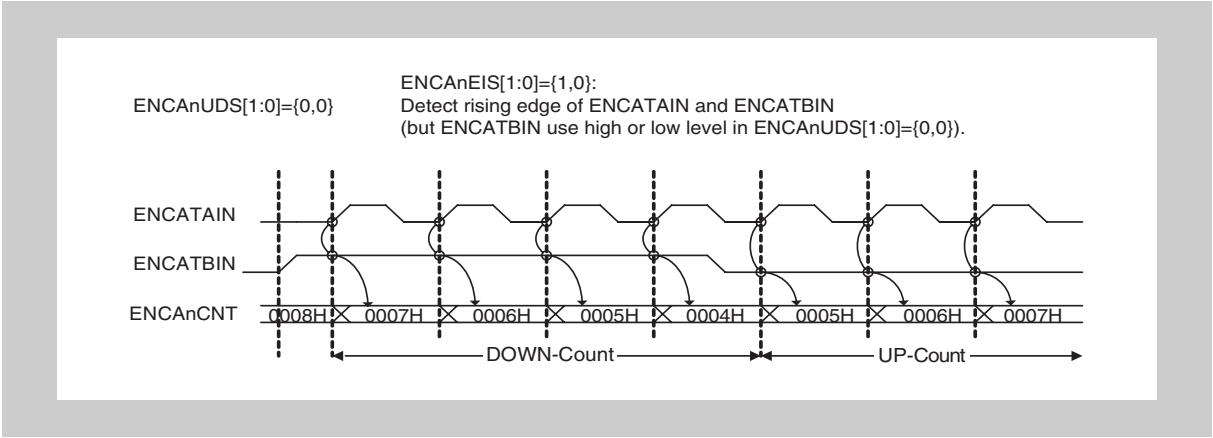


Figure 19-3 Count operation when ENCAnUDS1 and ENCAnUDS0 = {0, 0}

(2) When ENCAnUDS1 and ENCAnUDS0 = {0, 1}

ENCAnUDS1	ENCAnUDS0	Operation description		
		ENCATAIN input	ENCTBIN input	Count operation
0	1	Low level	Rising edge	Down
			Falling edge	
			Both edges	
		High level	Rising edge	
			Falling edge	
			Both edges	
		Rising edge	Low level	Up
		Falling edge		
		Both edges		
		Rising edge	High level	
		Falling edges		
		Both edges		
		Simultaneous input		Hold

The valid edges for ENCATAIN and ENCATBIN are specified by setting ENCAnEIS1 and ENCAnEIS0.

Count operation is performed when the valid edges of ENCATAIN and ENCATBIN overlap.

The following timing chart shows the count operation when ENCAnUDS1 and ENCAnUDS0 = {0, 1}.

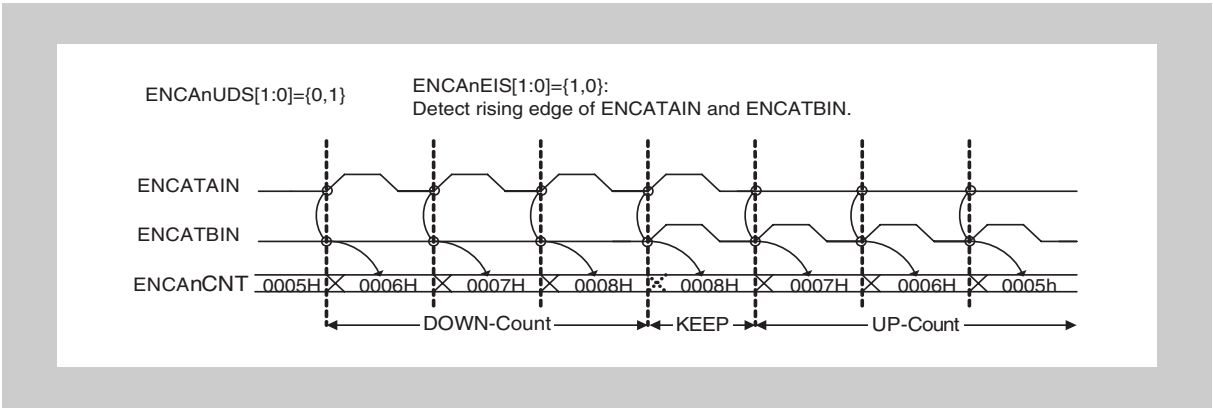


Figure 19-4 Count operation when ENCAnUDS1 and ENCAnUDS0 = {0, 1}

(3) When ENCAnUDS1 and ENCAnUDS0 = {1, 0}

ENCAnUDS1	ENCAnUDS0	Operation description		
		ENCATAIN input	ENCTBIN input	Count operation
1	0	Rising edge	Low level	Down
		Rising edge	Falling edge	
		Falling edge	Low level	Up
		Falling edge	Falling edge	
		Low level	Rising edge	Hold
		Rising edge	Rising edge	
		High level	Rising edge	
		Falling edge	Rising edge	
		Low level	Falling edge	
		Rising edge	High level	
		High level	Falling edge	
		Falling edge	High level	

Valid edge specification for ENCATAIN and ENCATBIN (settings of ENCAnEIS1 and ENCAnEIS0) is invalid.

The following timing chart shows the count operation when ENCAnUDS1 and ENCAnUDS0 = {1, 0}.

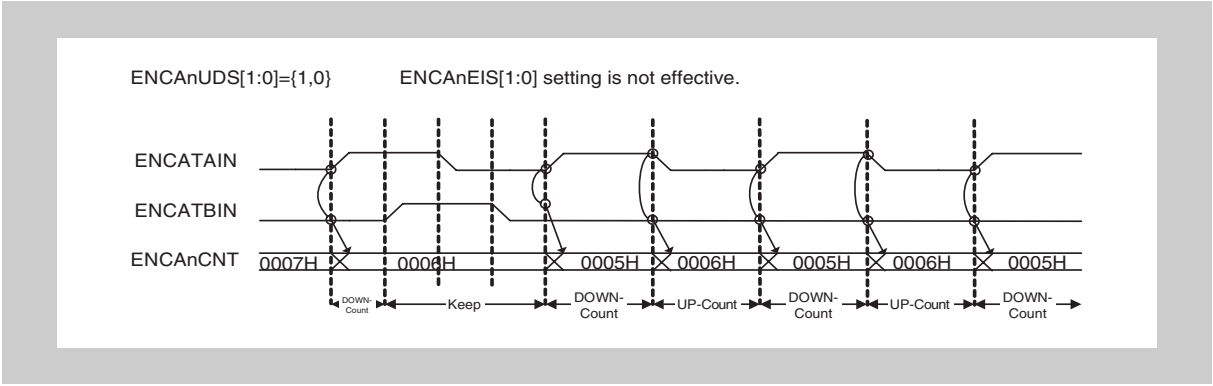


Figure 19-5 Count operation when ENCAnUDS1 and ENCAnUDS0 = {1, 0}

(4) When ENCAnUDS1 and ENCAnUDS0 = {1, 1}

ENCAnUDS1	ENCAnUDS0	Operation description		
		ENCATAIN input	ENCTBIN input	Count operation
1	1	Low level	Falling edge	Down
		Rising edge	Low level	
		High level	Rising edge	
		Falling edge	High level	
		Rising edge	High level	Up
		High level	Falling edge	
		Falling edge	Low level	
		Low level	Rising edge	
		Simultaneous input		Hold

Valid edge specification for ENCATAIN and ENCATBIN (settings of ENCAnEIS1 and ENCAnEIS0) is invalid.

The count value is held when the valid edges of ENCATAIN and ENCATBIN overlap.

The following timing chart shows the count operation when ENCAnUDS1 and ENCAnUDS0 = {1, 1}.

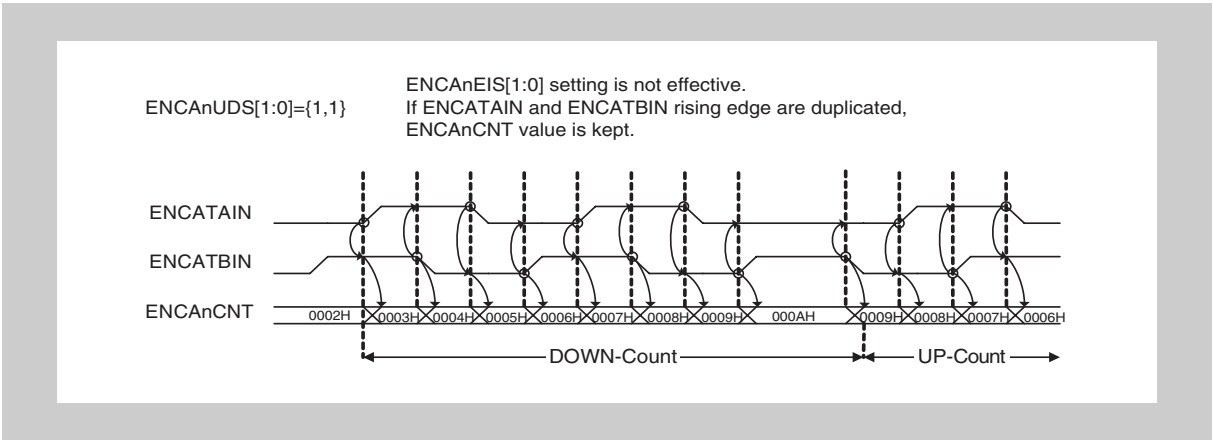


Figure 19-6 Count operation when ENCAnUDS1 and ENCAnUDS0 = {1, 1}

19.4.3 Timer counter clear control by encoder input

The timer counter is cleared to 0000H by phase Z encoder input (ENCATZIN).

Two types of clearing methods can be selected by controlling the ENCAnSCE, ENCAnZCL, ENCAnBCL, ENCAnACL, ENCAnECS1, and ENCAnECS0 bits of the ENCAnIOC1 register.

Clearing method	ENCAnSCE	ENCAnZCL	ENCAnBCL	ENCAnACL	ENCAnECS1, ENCAnECS0
(1)	0	Invalid	Invalid	Invalid	Valid
(2)	1	Valid	Valid	Valid	Invalid

(1) Clearing method when ENCAnSCE = 0

- Upon detection of the valid edge of ENCATZIN, the timer counter is cleared to 0000_H in synchronization with the operation clock.
- The valid edge of ENCATZIN is specified by the setting of the ENCAnECS1 and ENCAnECS0 bits.
- The settings of the ENCAnZCL, ENCAnBCL, and ENCAnACL bits are invalid.
- A clear interrupt signal (ENCATIEC) is output simultaneously with timer counter clearing.

(2) Clearing method when ENCAnSCE = 1

- The clear levels of the ENCATZIN, ENCATBIN, ENCATAIN inputs are detected according to the settings of the ENCAnZCL, ENCAnBCL, and ENCAnACL bits, and the timer counter is cleared to 0000_H in synchronization with the operating clock.
- The settings of the ENCAnECS1 and ENCAnECS0 bits are invalid.
- An encoder clear interrupt signal (ENCATIEC) is output simultaneously with timer counter clearing.

The clearing conditions of the timer counter according to the ENCAnZCL, ENCAnBCL, and ENCAnACL settings are listed in the table below.

Counter clear condition setting			Encoder input level		
ENCAnZCL	ENCAnBCL	ENCAnACL	ENCATZIN	ENCATBIN	ENCATAIN
0	0	0	Low	Low	Low
0	0	1	Low	Low	High
0	1	0	Low	High	Low
0	1	1	Low	High	High
1	0	0	High	Low	Low
1	0	1	High	Low	High
1	1	0	High	High	Low
1	1	1	High	High	High

19.4.4 Functions of ENCAnCCR0

(1) Compare function

- When ENCAnCRM0 = 0, the ENCAnCCR0 register functions as a dedicated compare register.
- Upon compare match between the value of the timer counter and the ENCAnCCR0 setting value, a compare 0 match interrupt (ENCATINT0) is output.
- When ENCAnECM0 = 1, the timer counter is cleared to 0000_H in synchronization with the operating clock upon compare match if the next count operation is up-count.

ENCAnCCR0 function	Compare match clear control	Next count operation	Timer counter clearing upon compare match with ENCAnCCR0
ENCAnCRM0	ENCAnECM0		
0 (Compare)	0	Up-count	Does not clear (continues count operation).
		Down-count	
	1	Up-count	Clears timer counter to 0000_H.
		Down-count	Does not clear (continues count operation).

When ENCAnLD = 1

- Upon occurrence of an underflow, the setting value of the ENCAnCCR0 register is loaded to the timer counter.
- An underflow interrupt (ENCATIUD) is output.

(2) Capture function

- When ENCAnCRM0 = 1, the ENCAnCCR0 register functions as a dedicated capture register.
- Upon valid edge detection of the capture trigger input 0 (ENCATTIN0), the value of the timer counter is stored into ENCAnCCR0.
- A capture 0 interrupt (ENCATINT0) is output during capture operation.

19.4.5 Functions of ENCAnCCR1

(1) Compare function

- When ENCAnCRM1 = 0, the ENCAnCCR1 register functions as a dedicated compare register.
- Upon compare match between the value of the timer counter and the ENCAnCCR1 setting value, a compare 1 match interrupt (ENCATINT1) is output.
- When ENCAnECM1 = 1, the timer counter is cleared to 0000_H in synchronization with the operating clock upon compare match if the next count operation is down-count.

ENCAnCCR1 function	Compare match clear control	Next count operation	Timer counter clearing upon compare match with ENCAnCCR1
ENCAnCRM1	ENCAnECM1		
0 (Compare)	0	Up-count	Does not clear (continues count operation).
		Down-count	
	1	Up-count	Does not clear (continues count operation).
		Down-count	Clears timer counter to 0000_H.

Compare match interrupt detection mask function

- When ENCAnCME = 1, the compare 1 match interrupt detection mask function is enabled. In this state, the compare 1 match interrupt is output upon the first match of the value of the timer counter and the ENCAnCCR1 setting value, and interrupts are then masked for the second and subsequent compare matches.
- When ENCAnMCS = 0, the compare 1 match interrupt detection mask function is disabled by a write operation to the ENCAnCCR1 register.
- When ENCAnMCS = 1, the compare 1 match interrupt detection mask function is disabled by a timer counter clear operation accompanying phase Z or by a timer counter clear operation upon match between the ENCAnCCR0 register value and the timer counter value.
- When ENCAnMCS = 1 and ENCAnLDE = 1, the compare 1 match interrupt detection mask function is disabled by a loading operation of the ENCAnCCR0 register to the timer counter upon underflow detection.
- Setting ENCAnECM to 1 is prohibited when enabling the compare 1 match interrupt detection mask function.

ENCAnCCR1 function	Compare 1 match interrupt mask	Interrupt mask cancel trigger		Compare 1 match interrupt output upon compare match with ENCAnCCR1
ENCAnCRM1	ENCAnCME	ENCAnMCS	Underflow occurrence upon ENCAnLDE = 0	
0 (Compare)	0 (Mask function disabled)	– (Setting invalid)	-	Outputs compare 1 match interrupt upon each compare match.
	1 (Mask function enabled)	0 (Write operation to ENCAnCCR1)	Occurred (Loading of ENCAnCCR0 to timer counter)	Outputs compare 1 match interrupt once upon the first compare match. (Interrupts are masked for the second and subsequent matches until the cancel trigger occurs.)
		1 (Timer counter clear operation)		

(2) Capture function

- When ENCAnCRM1 = 1, the ENCAnCCR1 register functions as a dedicated capture register.

The operations for each of the ENCAnCTS settings are shown in the table below.

ENCAnCCR1 function	Capture trigger selection	Capture trigger signal	Timer counter clearing	Interrupt occurrence
ENCAnCRM1	ENCAnCTS			
1 (Capture)	0	Capture trigger 1 input (ENCATTIN1)	Does not clear timer counter.	(1) Capture 1 interrupt (ENCATINT1)
	1	Encoder clear input accompanying phase Z (ENCATZIN)	Clears timer counter.	(1) Capture 1 interrupt (ENCATINT1) (2) Encoder clear interrupt (ENCATIEC)

(3) Timer counter clearing upon compare register match

Timer counter clearing upon compare match between the value of the timer counter and the ENCAnCCR0/1 setting value, according to the settings of the ENCAnECM1 and ENCAnECM0 bits, is detailed in the following table.

ENCAnECM1 and ENCAnECM0	Next count operation	Timer counter clearing upon compare match with ENCAnCCR1	Timer counter clearing upon compare match with ENCAnCCR0
00	Up-count	Does not clear (continues count operation).	Does not clear (continues count operation).
	Down-count	Does not clear (continues count operation).	Does not clear (continues count operation).
01	Up-count	Does not clear (continues count operation).	Clears timer counter to 0000_H.
	Down-count	Does not clear (continues count operation).	Does not clear (continues count operation).
10	Up-count	Does not clear (continues count operation).	Does not clear (continues count operation).
	Down-count	Clears timer counter to 0000_H.	Does not clear (continues count operation).
11	Up-count	Does not clear (continues count operation).	Clears timer counter to 0000_H.
	Down-count	Clears timer counter to 0000_H.	Does not clear (continues count operation).

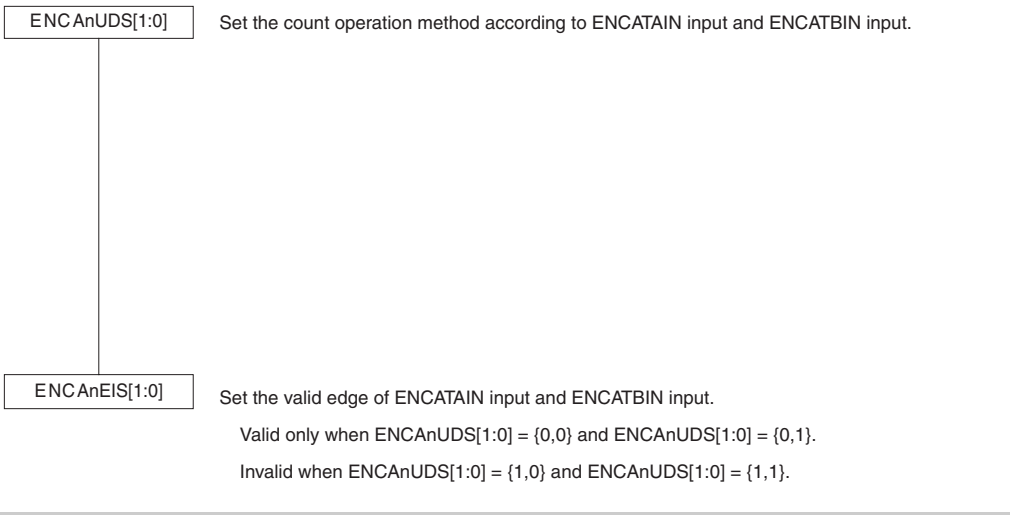
19.5 Setting Sequences

19.5.1 Encoder timer setting procedure

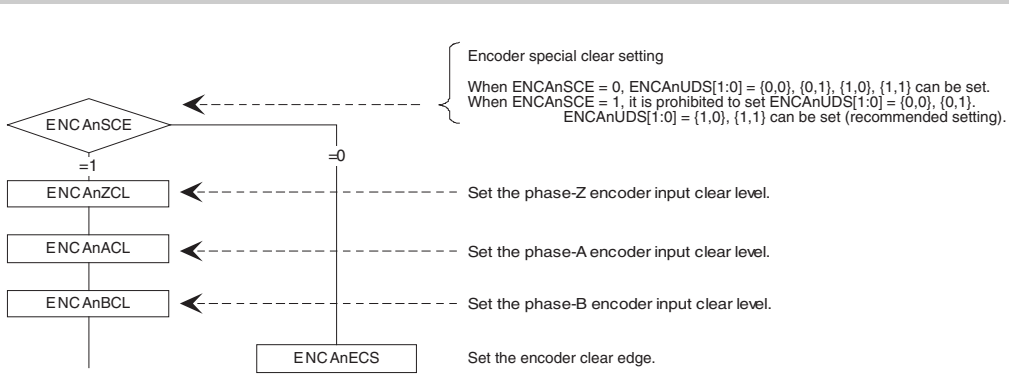
The encoder timer setting procedure is described below.

	Action	Setting status
Initial setting		Power-off status (Writing to each register is disabled)
	Reset release	Power-on status, ENCAn operation stopped status. (Writing to each register is enabled)
ENCA initial settings	Perform the following initial settings. • Setting for counter • Setting for counter clear • Setting for ENCAnCCR0 register • Setting for ENCAnCCR1 register	This is the count operation stopped status. The value of the ENCAnTE bit indicating the operating status is 0.
	Perform the counter initial value settings. • Set any 16-bit value to ENCAnCNT register. (When, after setting this register, the ENCAnTS bit is set to 1, the counter operation starts from the set count value.)	The set value is set as the initial value of the counter register.
Operation start	Perform the counter operation start setting. • Set the ENCAnTS bit to 1.	This is the counter operation starts status. The value of the ENCAnTE bit indicating the operating status is 1, and the count clock is supplied to the internal circuit.
Operating	Only those registers whose setting can be changed during operation can be rewritten. • ENCAnCCR0 register setting can be changed. • ENCAnCCR1 register setting can be changed. • ENCAnIOC0 register setting can be changed.	The count operation set with the initial setting is performed, and up/down counting is performed according to ENCATAIN and ENCATBIN.
Operation stop	Perform the counter operation stop setting during operation. • Set the ENCAnTT bit to 1.	This is the counter operation stopped status. The value of the ENCAnTE bit indicating the operating status is 0.
ENCA stop	Reset	This is the power-off status. The entire circuit and all the setting registers are initialized.

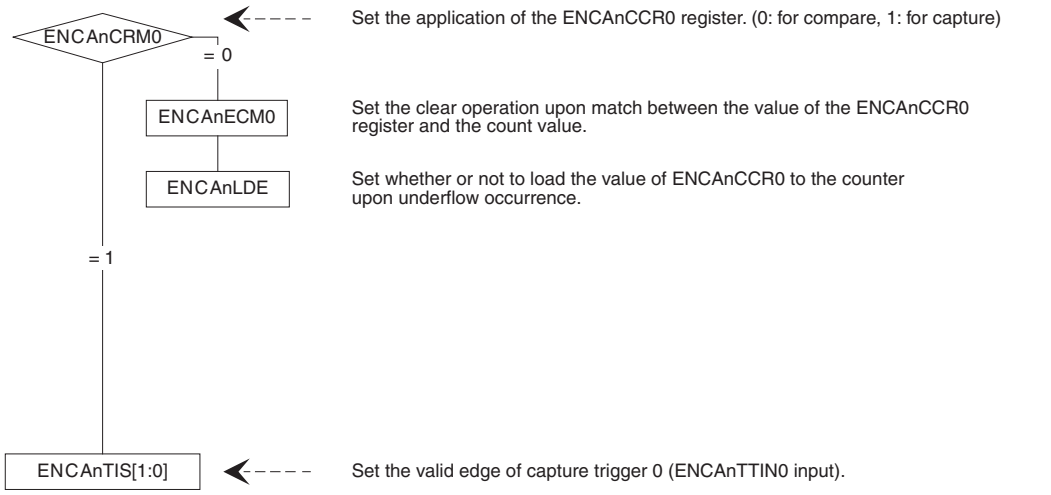
(1) Initial setting procedure for counter



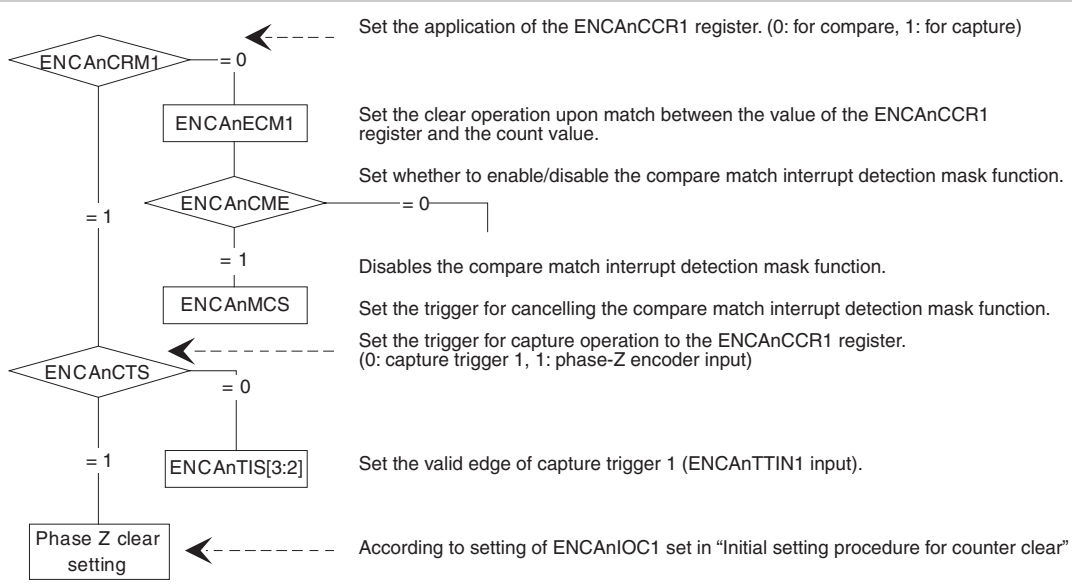
(2) Initial setting procedure for counter clear



(3) Setting procedure for ENCAnCCR0 register



(4) Setting procedure for ENCAnCCR1 register



Chapter 20 Asynchronous Serial Interface E (UARTE)

This chapter describes asynchronous serial interface E (UARTE).

The first section describes all properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names. The subsequent sections describe the features that apply to all implementations.

20.1 V850E2/Sx4-H UARTEn Features

Instances

This microcontroller has the following number of instances of UARTE:

<R> **Table 20-1 Instances of UARTE**

UARTE	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of instances	4	5	5
Name	UARTE0 to UARTE3	UARTE0 to UARTE3, UARTE10	UARTE0 to UARTE3, UARTE10

Instances index n

Throughout this chapter, the instance of UARTE is identified by the index "n" (n = 0 to 3, 10), for example, URTEnCTL0 for the UARTEn control register 0.

Register addresses

All UARTEn register addresses are given as addresses offset fromthe individual base address <URTEn_base> for UARTEn.
The <URTEn_base> address of UARTEn are given in the following table:

Table 20-2 Register base addresses <URTEn_base>

UARTEn	<UARTEn_base> address
UARTE0	FF5C 0000$_H$
UARTE1	FF5D 1000$_H$
UARTE2	FF5E 2000$_H$
UARTE3	FF5F 3000$_H$
UARTE10	FF66 5000$_H$

Clock supply The following clock is supplied to UARTEn:

Table 20-3 UARTEn clock supply

UARTEn	Clock	Connected to:
UARTE0	PCLK	Clock generator CKSCLK_112
UARTE1	PCLK	Clock generator CKSCLK_112
UARTE2	PCLK	Clock generator CKSCLK_114
UARTE3	PCLK	Clock generator CKSCLK_114
UARTE10	PCLK	Clock generator CKSCLK_011

Interrupts UARTEn can generate the interrupts requests below, which are input to their respective LIN master controller (LMAn).

Table 20-4 UARTE interrupt requests

UARTEn signal	Function	Connected to:
INTUAEnTIT	Transmission interrupt	Interrupt controller INTUAEnTIT
INTUAEnTIR	Transmission interrupt	Interrupt controller INTUAEnTIR
INTUAEnTIS	Status interrupt	Interrupt controller INTUAEnTIS
INTUAEnTRA	Receive/status interrupt	Not connected

URTE hardware reset UARTE and its registers are initialized by the following reset signal:

Table 20-5 URTEn reset signal

URTEn	Reset signal
URTEn	System reset SYSRES

I/O signals The I/O signals of UARTE are listed in the following table:

Table 20-6 URTEn I/O signals

URTEn signals	Function	Connected to:
URTE0:		
URTE0TTXD	Transmit data output	Port URTE0RX pin
URTE0TRXD	Receive data input	Port URTE0TX pin
URTE1:		
URTE1TTXD	Transmit data output	Port URTE1RX pin
URTE1TRXD	Receive data input	Port URTE1TX pin
URTE2:		
URTE2TTXD	Transmit data output	Port URTE2RX pin
URTE2TRXD	Receive data input	Port URTE2TX pin
URTE3:		
URTE3TTXD	Transmit data output	Port URTE1RX pin
URTE3TRXD	Receive data input	Port URTE1TX pin
URTE10:		
URTE10TTXD	Transmit data output	Port URTE10RX pin
URTE10TRXD	Receive data input	Port URTE10TX pin

<R> **Caution** A port filter is assigned to the receive data input pin (URTEnRX) of asynchronous serial interface E (UARTEn), and this filter is enabled as the initial setting. However, because UARTEn has an internal filter, do not use the port filter when using UARTEn; instead, enable the filter bypass by setting the corresponding registers as shown below.

URTE0RX: FCLA26CTL4 = 80H, URTE1RX: FCLA26CTL5 = 80H, URTE2RX: FCLA27CTL0 = 80H

URTE3RX: FCLA27CTL1 = 80H, URTE10RX: FCLA7CTL0 = 80H

Baudrate measurement The following URTEn receive data signals can be internally connected to a capture input of timer array unit A.

Table 20-7 URTEn timer connections

URTEn signals	Function	Connected to:
URTE10:		
Port URTE10RX	Receive data input	TAUA0 TAUA0TTIN0

Note See *15.2 "TAUA Input Selection"* for details.

20.2 Features

- Full-duplex communication:
  - Internal UARTEn receive data register n (URTEnRX)
  - Internal UARTEn transmit data register n (URTEnTX)
- 2-pin configuration:
  - URTEnTTXD: Transmit data output pin
  - URTEnTRXD: Receive data input pin
- Reception error and status output function
  - Parity error
  - Framing error
  - Overrun error
  - Data consistency error
- Interrupt requests: 4
  - Transmission interrupt INTUAEnTIT
  - Reception interrupt INTUAEnTIR
  - Status interrupt INTUAEnTIS
  - Receive/status interrupt INTUAEnTRA
- Character length: 7, 8 bits
- Parity function: odd, even, 0, none
- Transmission stop bit: 1, 2 bits
- MSB-/LSB-first transfer selectable
- Transmit/receive data inverted input/output possible
- 13 to 20 bits selectable for the BF (Break Field) in the LIN (Local Interconnect Network) communication format
  - Recognition of 11 bits or more possible for BF reception in LIN communication format
  - BF reception flag provided
- BF reception can be detected during data communication
- Bus monitor function to keep data consistency of the transmit data

20.3 Configuration

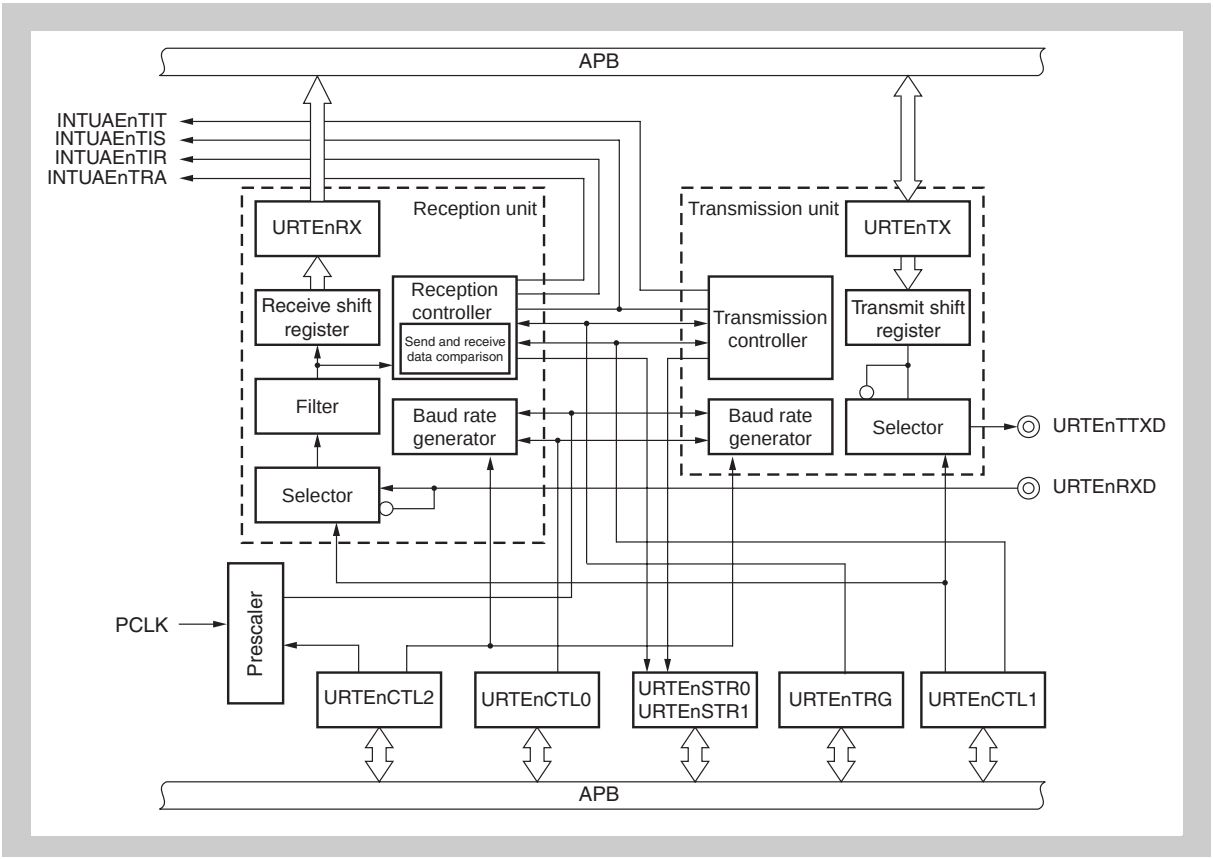


Figure 20-1 Block diagram of Asynchronous Serial Interface UARTEn

20.4 UARTEn Registers

The UARTEn is controlled and operated by the following registers:

Table 20-8 UARTEn registers

Register function	Name	Address
Control register 0	URTEnCTL0	<URTEn_base> + 00_H
Control register 1	URTEnCTL1	<URTEn_base> + 20_H
Control register 2	URTEnCTL2	<URTEn_base> + 24_H
Trigger register	URTEnTRG	<URTEn_base> + 04_H
Status register 0	URTEnSTR0	<URTEn_base> + 08_H
Status register 1	URTEnSTR1	<URTEn_base> + $0C_H$
Status clear register	URTEnSTC	<URTEn_base> + 10_H
Receive data register	URTEnRX	<URTEn_base> + 14_H
Transmit data register	URTEnTX	<URTEn_base> + 18_H

<URTEn_base> The base addresses <URTEn_base> of the UARTEn are defined in the first section of this chapter under the key word “Register addresses”.

(1) URTEnCTL0 - UARTEn control register 0

This register controls the serial transfer operation of the UARTEn.

Access This register can be read or written in 8-bit and 1-bit units.

Address <URTEn_base> + 00_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
URTEn PW	URTEn TXE	URTEn RXE	0	0	0	0	URTEn SLDC
R/W	R/W	R/W	R	R	R	R	R/W

Table 20-9 URTEnCTL0 register contents

Bit position	Bit name	Function
7	URTEnPW	UARTEn enable 0: Stop UARTEn operation 1: Enable UARTEn operation Changing this bit initializes all transmission and reception units.
6	URTEnTXE	Transmission operation enable 0: Disable transmission operation 1: Enable transmission operation • To start transmission, set URTEnPW, and then set URTEnTXE. To stop transmission, clear URTEnTXE, and then clear URTEnPW (they can be cleared at the same time). • To initialize the transmission unit, clear URTEnTXE, wait for two prescaler clock cycles, and then set URTEnTXE again. For details about the prescaler clock, see *(3) "URTEnCTL2 - UARTEn control register 2"*.
5	URTEnRXE	Reception operation enable 0: Disable reception operation 1: Enable reception operation • To enable reception, set URTEnPW, and then set URTEnRXE. To stop reception, clear URTEnRXE, and then clear URTEnPW (they can be cleared at the same time). • To initialize the reception unit, clear URTEnRXE, wait for two prescaler clock cycles, and then set URTEnRXE again. Reception is enabled when the time of two prescaler clock cycles has elapsed since URTEnRXE is set. The rising edge detection of the URTEnTRXD signal is enabled when four prescaler clock cycles has elapsed after URTEnRXE is set. For details about the prescaler clock, see *(3) "URTEnCTL2 - UARTEn control register 2"*.
0	URTEnSLDC	Data consistency check enable 0: Disable consistency check 1: Enable consistency check This bit selects the handling of data consistency error checks when transmitting data. When this bit is set to 1, the transmit data and receive data are compared, and if a mismatch is detected, URTEnSTR1.URTEnDCE is set to 1 and a status interrupt request INTUAEnTIS is issued. This bit is referenced only when starting transmission. Consequently, if this bit value is changed later on during transmission processing, the transmission processing continues, using the value set at the start of transmission.

Cautions

1. Disable transmission if UARTEn meets all the conditions below:
   - Transmission and reception are enabled (URTEnCTL0.URTEnPW = URTEnRXE = URTEnTXE = 1).
   - Data consistency check is enabled (URTEnCTL0.URTEnSLDC = 1).
   - Data is being transmitted or has been transmitted.

   Use the following procedure to keep reception enabled:
   - Check that no data is pending for transmission (URTEnSTR0.URTEnSSBT = URTEnSST = 0).
   - Check that no data is pending for reception (URTEnSTR0.URTEnSSBR = URTEnSSR = 0).
   - Disable transmission by clearing URTEnCTL0.URTEnTXE.

   The reason why this procedure is required is that the data consistency error flag URTEnSTR1.URTEnDCE is cleared if URTEnCTL0.URTEnTXE is cleared.
   Thus a potential data consistency error would not occur if transmission is disabled during a data transfer or after its completion.
2. Disable reception if UARTEn meets all the conditions below:
   - Transmission and reception are enabled (URTEnCTL0.URTEnPW = URTEnRXE = URTEnTXE = 1).
   - Data consistency check is enabled (URTEnCTL0.URTEnSLDC = 1).
   - Data is being transmitted or has been transmitted.

   Use the following procedure to keep transmission enabled:
   - Check that no data is pending for transmission (URTEnSTR0.URTEnSSBT = URTEnSST = 0).
   - Check that no data is pending for reception (URTEnSTR0.URTEnSSBR = URTEnSSR = 0).
   - Disable reception by clearing URTEnCTL0.URTEnRXE.

   The reason why this procedure is required is that the data consistency error flag URTEnSTR1.URTEnDCE is cleared and invalid if URTEnCTL0.URTEnTXE is cleared.
   Thus a potential data consistency error of already transmitted data would not occur.
3. Do not start data transmission if all the conditions below are met:
   - Data consistency check is enabled (URTEnCTL0.URTEnSLDC = 1).
   - BF reception is enabled (URTEnSTR0.URTEnSSBR = 1).
   - BF detection during reception is disabled (URTEnCTL1.URTEnSLBM = 0).

   A data consistency error will occur under above conditions when BF reception is completed. The status interrupt INTUAEnTIS will be asserted and the reception interrupt request INTUAEnTIR will not be generated (URTEnSTR1.URTEnBSF remains 0). Consequently BF reception completion will not be recognized.

(2) URTEnCTL1 - UARTEn control register 1

This register defines the data frame properties of the UARTEn serial data transfers.

Access This register can be read or written in 16-bit units.

Address <URTEn_base> + 20_H

Initial Value 5002_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
URTEn SLBM	URTEnBLG[2:0]			0	0	0	URTEn CLG
R/W	R/W	R/W	R/W	R	R	R	R/W

7	6	5	4	3	2	1	0
URTEnSLP[1:0]		URTEn TDL	URTEn RDL	0	URTEn SLG	URTEn SLD	URTEn SLIT
R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

Table 20-10 URTEnCTL1 register contents (1/3)

Bit position	Bit name	Function
15	URTEnSLBM	BF receive mode selection 0: BF reception during data reception disabled. 1: BF reception during data reception enabled. • Changing this bit is only allowed if reception is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = 0).
14 to 12	URTEn BLG[2:0]	BF bit length during transmission URTEnBLG2 / URTEnBLG1 / URTEnBLG0 / BF length: 1 0 1 13 bits 1 1 0 14 bits 1 1 1 15 bits 0 0 0 16 bits 0 0 1 17 bits 0 1 0 18 bits 0 1 1 19 bits 1 0 0 20 bits Changing these bits is only allowed if transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnTXE = 0).
8	URTEnCLG	Receive/transmit data bit length 0: 7 bits 1: 8 bits • When the transmission/reception is performed in the LIN format, set URTEnCLG to 1. • Changing this bit is only allowed if reception and transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = 0 and URTEnCTL0.URTEnTXE = 0).

Table 20-10 URTEnCTL1 register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7, 6</td><td>URTEn SLP[1:0]</td><td>Parity bit selection
<table>
<tr><th rowspan="2">URTEnSLP1</th><th rowspan="2">URTEnSLP0</th><th colspan="2">Operation</th></tr>
<tr><th>Transmission</th><th>Reception</th></tr>
<tr><td>0</td><td>0</td><td>Output without parity bit</td><td>Received with no parity</td></tr>
<tr><td>0</td><td>1</td><td>Output 0 parity (0-fixed)</td><td>No parity judgment</td></tr>
<tr><td>1</td><td>0</td><td>Output odd parity</td><td>Judged as odd parity</td></tr>
<tr><td>1</td><td>1</td><td>Output even parity</td><td>Judged as even parity</td></tr>
</table>
• If “Reception with no parity judgment” is selected during reception, a parity check is not performed. Therefore, because the URTEnSTR1.UTREnPE bit is not set, no error interrupt is output.
• When transmission/reception is performed in the LIN format, the URTEnPE bit is not set and no error interrupt is output upon detection of a parity error.
• Changing these bits is only allowed if reception and transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = URTEnCTL0.URTEnTXE = 0).</td></tr>
<tr><td>5</td><td>URTEnTDL</td><td>Transmission data level control
0: No inverted output of transmit data
1: Inverted output of transmit data
• The output level of the URTEnTTXD pin can be inverted using this bit. It inverts the URTEnTTXD output level immediately, regardless of the values of URTEnCTL0.URTEnPW and URTEnCTL0.URTEnTXE. Therefore, if URTEnTDL is set to 1 while the operation is disabled, the URTEnTTXD outputs low level.
• Changing this bit is only allowed if transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnTXE = 0).</td></tr>
<tr><td>4</td><td>URTEnRDL</td><td>Reception data level control
0: No inverted output of receive data
1: Inverted output of receive data
• The output level of the URTEnTRXD pin can be inverted using this bit. It inverts the URTEnTRXD input level immediately, regardless of the values of URTEnCTL0.URTEnPW and URTEnCTL0.URTEnRXE. Therefore, if URTEnRDL is set to 1 while the operation is disabled, the URTEnTRXD inputs low level.
• Changing this bit is only allowed if reception is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = 0).</td></tr>
</table>

Table 20-10 URTEnCTL1 register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7, 6</td><td>URTEn SLP[1:0]</td><td>Parity bit selection
<table>
<tr><th rowspan="2">URTEnSLP1</th><th rowspan="2">URTEnSLP0</th><th colspan="2">Operation</th></tr>
<tr><th>Transmission</th><th>Reception</th></tr>
<tr><td>0</td><td>0</td><td>Output without parity bit</td><td>Received with no parity</td></tr>
<tr><td>0</td><td>1</td><td>Output 0 parity (0-fixed)</td><td>No parity judgment</td></tr>
<tr><td>1</td><td>0</td><td>Output odd parity</td><td>Judged as odd parity</td></tr>
<tr><td>1</td><td>1</td><td>Output even parity</td><td>Judged as even parity</td></tr>
</table>
• If “Reception with no parity judgment” is selected during reception, a parity check is not performed. Therefore, because the URTEnSTR1.UTREnPE bit is not set, no error interrupt is output.
• When transmission/reception is performed in the LIN format, the URTEnPE bit is not set and no error interrupt is output upon detection of a parity error.
• Changing these bits is only allowed if reception and transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = URTEnCTL0.URTEnTXE = 0).</td></tr>
<tr><td>5</td><td>URTEnTDL</td><td>Transmission data level control
0: No inverted output of transmit data
1: Inverted output of transmit data
• The output level of the URTEnTTXD pin can be inverted using this bit. It inverts the URTEnTTXD output level immediately, regardless of the values of URTEnCTL0.URTEnPW and URTEnCTL0.URTEnTXE. Therefore, if URTEnTDL is set to 1 while the operation is disabled, the URTEnTTXD outputs low level.
• Changing this bit is only allowed if transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnTXE = 0).</td></tr>
<tr><td>4</td><td>URTEnRDL</td><td>Reception data level control
0: No inverted output of receive data
1: Inverted output of receive data
• The output level of the URTEnTRXD pin can be inverted using this bit. It inverts the URTEnTRXD input level immediately, regardless of the values of URTEnCTL0.URTEnPW and URTEnCTL0.URTEnRXE. Therefore, if URTEnRDL is set to 1 while the operation is disabled, the URTEnTRXD inputs low level.
• Changing this bit is only allowed if reception is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = 0).</td></tr>
</table>

Table 20-10 URTEnCTL1 register contents (3/3)

Bit position	Bit name	Function
2	URTEnSLG	Stop bit number selection for transmission data 0: 1 bit 1: 2 bits • The stop bit length during data or BF reception is always handled as "1". • Changing this bit is only allowed if transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnTXE = 0).
1	URTEnSLD	Transfer direction selection 0: MSB-first transfer 1: LSB-first transfer • The stop bit length during data or BF reception is always handled as "1". • When the transmission/reception is performed in the LIN format, set URTEnSLD to 1. • Changing this bit is only allowed if reception and transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = URTEnCTL0.URTEnTXE = 0).
0	URTEnSLIT	Transmission interrupt request (INTUAEnTIT) timing selection 0: INTUAEnTIT generated at the start of transmission, i.e. when the transmit data is stored to the transmission shift register 1: INTUAEnTIT generated at transmission completion • Changing this bit is only allowed if transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnTXE = 0).

(3) URTEnCTL2 - UARTEn control register 2

This register defines the baud rates of the UARTEn serial data transfers.

Access This register can be read or written in 16-bit units.

Address <URTEn_base> + 24_H

Initial Value $EFFF_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
URTEnPRS[2:0]			0	URTEnBRS[11:8]			
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
URTEnBRS[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-11 URTEnCTL2 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15 to 13</td><td>URTEn
PRS[2:0]</td><td>Prescaler clock (PRSCLK) division value
0: PRSCLK = PCLK / 2^0
1: PRSCLK = PCLK / 2^1
2: PRSCLK = PCLK / 2^2
3: PRSCLK = PCLK / 2^3
4: PRSCLK = PCLK / 2^4
5: PRSCLK = PCLK / 2^5
6: PRSCLK = PCLK / 2^6
7: PRSCLK = PCLK / 2^7</td></tr>
<tr><td>11 to 0</td><td>URTEn
BRS[11:0]</td><td>Baud rate clock (BRCLK) division value
<table>
<tr><th>URTEn BRS[11:0]</th><th>Transmit/receive BRCLK</th><th>BF receive clock</th></tr>
<tr><td>000_H</td><td rowspan="5">PCLK / (2 x 4)</td><td rowspan="5">PCLK / 4</td></tr>
<tr><td>001_H</td></tr>
<tr><td>002_H</td></tr>
<tr><td>003_H</td></tr>
<tr><td>004_H</td></tr>
<tr><td>005_H</td><td>PCLK / (2 x 5)</td><td>PCLK / 5</td></tr>
<tr><td>...</td><td>PCLK /
(2 x URTEnBRS[11:0]</td><td>PCLK / URTEnBRS[11:0]</td></tr>
<tr><td>FFE_H</td><td>PCLK / (2 x 4094)</td><td>PCLK / 4094</td></tr>
<tr><td>FFF_H</td><td>PCLK / (2 x 4095)</td><td>PCLK / 4095</td></tr>
</table></td></tr>
</table>

Caution Writing to this register is only allowed if the UARTEn operation is disabled (URTEnCTL0.URTEnPW = 0).

PCLK The value of the UARTEn input clock is defined in the first section of this chapter under the key word “Clock supply”.

(4) URTEnTRG - UARTEn trigger register

This register controls the UARTEn transmission/reception trigger of BF.

Access This register can be read or written in 8-bit and 1-bit units.

Address <URTEn_base> + 04_H

Initial Value 0000H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	URTEnBRT	URTEnBTT	0	0	0	0	0
R	R/W	R/W	R	R	R	R	R

Table 20-12 URTEnTRG register contents

Bit position	Bit name	Function
6	URTEnBRT	BF reception trigger 0: Read value is always 0, writing 0 is ignored 1: Enable BF reception • When reception is enabled, writing 1 to this bit enables BF reception (URTEnSTR0.URTEnSSBR = 1) and BF reception processing begins when the falling edge of the receive serial signal is detected. • If 1 is written to this bit during reception processing, the current reception processing is terminated. Consequently, the received data is not stored, the framing, parity and overflow error bits are not updated based on the data that was being received and no interrupts are generated. Meanwhile, the BF counter value is continuously being used. • After BF reception, the reception status is set according to the URTEnCTL1.URTEnSLBM setting. • Setting this bit to 1 is only allowed if reception is enabled (URTEnCTL0.URTEnPW = URTEnCTL0.URTEnRXE = 1). After URTEnBRT is set to 1, completion of BF reception is reported by either of the following two methods, based on the URTEnCTL1.URTEnSLBM setting: • if URTEnCTL1.URTEnSLBM = 0: When BF reception is complete, a reception interrupt request INTUAEnTIR is generated. • if URTEnCTL1.URTEnSLBM = 1: When BF reception is complete, URTEnSTR1.URTEnBSF is set to 1 and the status interrupt request INTUAEnTIS is generated.
5	URTEnBTT	BF transmission trigger 0: Read value is always 0, writing 0 is ignored 1: Enable BF transmission • When this bit is set while URTEnSTR0.URTEnSSBT = 0 and transmission is enabled (URTEnDCE = 0), a BF transmit request is set, and URTEnSSBT is set to 1. • When this bit is set during data transmission, a BF is transmitted after the current transmission processing is completed. Even if this bit is set before the BF transmission is completed, a BF is transmitted only once. • When transmission is enabled (URTEnPW = URTEnTXE = 1), setting this bit clears all previously set data transmit requests (which have not been transmitted), leaving only BF transmit requests. If the URTEnTX7 to URTEnTX0 bits are written after setting this bit, data is transmitted after the BF is transmitted. • If both a BF transmit request and a data transmit request have been set when a transmission starts, the BF transmission takes priority. • When URTEnDCE = 1, writing 1 to this bit is ignored. • Setting this bit to 1 is only allowed if transmission is enabled (URTEnCTL0.URTEnPW = URTEnCTL0.URTEnTXE = 1).

(5) URTEnSTR0 - UARTEn status register 0

This register indicates the current status of serial data transmissions.

Access This register can be read in 8-bit or 1-bit units.
Writing to this register is only allowed if UARTEn operation is disabled (URTEnCTL0.URTEnPW = 0). If UARTEn operation is enabled (URTEnCTL0.URTEnPW = 1), any written values are disregarded and the initial values are restored.

Address <URTEn_base> + 08_H

Initial Value 0000_H. This register is initialized by any reset and when URTEnCTL0.URTEnPW is set or cleared.

7	6	5	4	3	2	1	0
0	URTEn SSBR[a]	URTEn SSBT[b]	0	0	0	URTEn SSR[a]	URTEn SST[b]
R	R	R	R	R	R	R	R

a) These bits are also initialized if reception is disabled by URTEnCTL0.URTEnRXE = 0.

b) These bits are also initialized if transmission is disabled by URTEnCTL0.URTEnTXE = 0.

Table 20-13 URTEnSTR0 register contents

Bit position	Bit name	Function
6	URTEnSSBR	BF reception enable status 0: BF reception is disabled 1: BF reception is enabled by setting URTEnTRG.URTEnBRT to 1 (BF reception standby mode or BF reception busy).
5	URTEnSSBT	BF transmission enable status 0: BF transmission is disabled 1: BF transmission is enabled by setting URTEnTRG.URTEnBTT to 1 (BF transmission standby mode or BF transmission busy).
1	URTEnSSR	Data reception status 0: No data reception ongoing 1: Data reception ongoing (data reception busy)
0	URTEnSST	Data transmission status 0: No transmission pending or ongoing 1: Data in URTEnTX[7:0] pending to be transmitted or transmission ongoing

(6) URTEnSTR1 - UARTEn status register 1

This register indicates results of serial data transmissions.

Access This register can be read in 8-bit and 1-bit units.
Writing to this register is only allowed if UARTEn operation is disabled (URTEnCTL0.URTEnPW = 0). If UARTEn operation is enabled (URTEnCTL0.URTEnPW = 1), any written values are disregarded and the initial values are restored.

Address <URTEn_base> + $0C_H$

Initial Value 0000_H. This register is initialized by any reset and when URTEnCTL0.URTEnPW is set or cleared.

7	6	5	4	3	2	1	0
0	0	0	URTEn BSF[a]	URTEn DCE[b]	URTEn PE[a]	URTEn FE[a]	URTEn OVE[a]
R	R	R	R	R	R	R	R

a) These bits are also initialized if reception is disabled by URTEnCTL0.URTEnRXE = 0.

b) This bit is also initialized if transmission is disabled by URTEnCTL0.URTEnTXE = 0.

Table 20-14 URTEnSTR1 register contents (1/2)

Bit position	Bit name	Function
4	URTEnBSF	BF reception successful flag 0: BF transmission is disabled by clearing URTEnTRG.URTEnBTT. 1: BF transmission is enabled by setting URTEnTRG.URTEnBTT (BF transmission standby mode or BF transmission busy). The URTEnBSF bit is cleared by the following: - URTEnCTL0.URTEnPW = 1 - URTEnCTL0.URTEnRXE = 0 - URTEnSTC.URTEnCLBS = 1
3	URTEnDCE	Data consistency error flag 0: Transmit/receive data (transmit/receive BF) mismatch was not detected. 1: Transmit/receive data (transmit/receive BF) mismatch was detected. When the BF receive mode selection bit is set during LIN communication, it is necessary to read this bit by using status interrupt processing and to confirm the beginning of a new frame slot. The URTEnDCE bit is cleared by the following: - URTEnCTL0.URTEnPW = 0 - URTEnCTL0.URTEnTXE = 0 - URTEnSTC.URTEnCLDC = 1
2	URTEnPE	Parity error flag 0: No parity error was detected in the received data. 1: A parity error was detected in the received data. The operation of URTEnPE is controlled by the settings of URTEn.URTEnSLP[1:0]. The URTEnPE bit is cleared by the following: - URTEnCTL0.URTEnPW = 0 - URTEnCTL0.URTEnRXE = 0 - URTEnSTC.URTEnCLP = 1
1	URTEnFE	Framing error flag 0: No framing error was detected in the received data. 1: A framing error was detected in the received data. The URTEnFE bit is cleared by the following: - URTEnCTL0.URTEnPW = 0 - URTEnCTL0.URTEnRXE = 0 - URTEnSTC.URTEnCLF = 1

Table 20-14 URTEnSTR1 register contents (2/2)

Bit position	Bit name	Function
0	URTEnOVE	Overrun error flag 0: No overrun error was detected in the received data. 1: An overrun error was detected in the received data. If an overrun error occurs, the data is discarded, and the next data received will not written to the receive data register URTEnRX. The URTEnOVE bit is cleared by the following: - URTEnCTL0.URTEnPW = 0 - URTEnCTL0.URTEnRXE = 0 - URTEnSTC.URTEnCLOV = 1

Note If the bits of these registers are set (1) and cleared (0) at the same time, setting takes priority over clearing.

For details about error detections, see *20.6.5 "Transmission data consistency check"* and *20.6.9 "Reception errors"*.

Caution In case reception and transmission is enabled and a consistency check error occurs (URTEnSTR1.URTEnDCE = 1), follow the procedure below prior next data transmission:

- disable transmission by URTEnCTL0.URTEnTXE = 0
- enable transmission by URTEnCTL0.URTEnTXE = 1
- initiate a transmission by URTEnTRG.URTEnBTT = 1 (BT transmission trigger) or writing any data to URTEnTX

Afterwards new transmissions can be started.

(7) URTEnSTC - UARTEn status clear register

This register is used to clear the status bits of the status register 1 URTEnSTR1.

Access This register can be read or written in 8-bit and 1-bit units.
Reading this register returns always 00_H.

Address <URTEn_base> $+10_H$

Initial Value 0000_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	URTEn CLBS	URTEn CLDC	URTEn CLP	URTEn CLF	URTEn CLOV
R	R	R	R/W	R/W	R/W	R/W	R/W

Table 20-15 URTEnSTC register contents

Bit position	Bit name	Function
4	URTEnCLBS	Clear BF reception successful flag 0: Writing 0 is ignored 1: Writing 1 clears URTEnSTR1.URTEnBSF
3	URTEnCLDC	Clear data consistency error flag 0: Writing 0 is ignored 1: Writing 1 clears URTEnSTR1.URTEnDCE If URTEnDCE is cleared by setting URTEnCLDC, any pending data or BF transmit requests will be ignored.
2	URTEnCLP	Clear parity error flag 0: Writing 0 is ignored 1: Writing 1 clears URTEnSTR1.URTEnPE
1	URTEnCLF	Clear framing error flag 0: Writing 0 is ignored 1: Writing 1 clears URTEnSTR1.URTEnFE
0	URTEnCLOV	Clear overrun error flag 0: Writing 0 is ignored 1: Writing 1 clears URTEnSTR1.URTEnOVE

(8) URTEnRX - UARTEn receive data register

This register stores received data.

The data stored in the receive shift register is transferred to URTEnRX upon completion of reception of 1 byte of data.

7-bit transfers

If the data length has been specified as 7 bits (URTEnCTL1.URTEnCLG = 0) and

- reception is LSB-first (URTEnCTL1.URTEnSLD = 1),:
  The receive data is transferred to URTEnRX[6:0] and the MSB URTEnRX[7] always becomes 0.
- reception is MSB-first (URTEnCTL1.URTEnSLD = 0),:
  The receive data is transferred to URTEnRX[7:1] and the LSB URTEnRX[0] always becomes 0.

For details about data formats, see *20.6.1 "Data formats"*.

Overrun error

When an overrun error (URTEnSTR1.URTEnOVE = 1) occurs, the receive data at this time is not transferred to URTEnRX and is discarded.

When reception processing ends and data reception is confirmed without any overrun errors, the received data is stored to URTEnRX according to the specified storage format.
Access

This register can be read in 8-bit units.
Writing to this register is only allowed if UARTEn operation is disabled (URTEnCTL0.URTEnPW = 0). If UARTEn operation is enabled (URTEnCTL0.URTEnPW = 1), any written values are disregarded and the initial values are restored.

Access

This register can be read in 8-bit units.

Address

<URTEn_base> + 14_H

Initial Value

FF_H. This register is initialized by any reset and when UARTEn operation is enabled by setting URTEnCTL0.URTEnPW.

7	6	5	4	3	2	1	0
URTEnRX[7:0]							
R	R	R	R	R	R	R	R

Table 20-16 URTEnRx register contents

Bit position	Bit name	Function
7 to 0	URTEn RX[7:0]	URTEn receive data

(9) URTEnTX - UARTEn transmit data register

This register is used to stores data to be transmitted.

Transmit data in URTEnTX is stored to the transmission shift register according to the specified transmit data format.

7-bit transfers If the data length has been specified as 7 bits (URTEnCTL1.URTEnCLG = 0) and

- transmission is LSB-first (URTEnCTL1.URTEnSLD = 1),: URTEnTX[6:0] is transferred to the shift register.
- transmission is MSB-first (URTEnCTL1.URTEnSLD = 0),: The LSB URTEnTX[0] is always set to “0” and URTEnTX[7:1] is transferred to the shift register.

For details about data formats, see *20.6.1 "Data formats"*.

Access This register can be read or written in 8-bit units.

Address <URTEn_base> + 18_H

Initial Value FF_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
URTEnTX[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-17 URTEnTx register contents

Bit position	Bit name	Function
7 to 0	URTEn TX[7:0]	URTEn transmit data

When transmission is enabled (URTEnCTL0 bits URTEnPW = URTETXE = 1), a write to URTEnTX starts transmission.

Note If the next data is written to URTEnTX before the current transmission is completed, continuous transmission can be enabled by waiting for the current data transmission to end before transmitting the next data.

20.5 Interrupt Request Signals

The following four interrupt request signals are generated by UARTEn.

- Transmission interrupt request INTUAEnTIT
- Reception interrupt request INTUAEnTIR
- Status interrupt request INTUAEnTIS
- Receive/status interrupt request INTUAEnTRA

20.5.1 Transmission interrupt request INTUAEnTIT

If transmit data is transferred from the URTEnTX register to transmit shift register with transmission enabled, the transmission interrupt request INTUAEnTIT is generated.

The condition for generation of a transmit interrupt request depends on the setting of the URTEnCTL1.URTEnSLIT:

- at start of transmission process: URTEnCTL1.URTEnSLIT = 0

A transmission interrupt request is issued when starting transmission of the first bit (this is the start bit for data transmission or the first BF bit for BF transmission).

- at end of transmission process: URTEnCTL1.URTEnSLIT = 1

A transmission interrupt request is issued after completing transmission of the last bit (the first bit of the stop bit when the stop bit length is 1, or the second bit of the stop bit when the stop bit length is 2).

Note If a data consistency error is detecting, this interrupt is not generated.

The following diagrams show the timing of the transmission interrupt request for both cases.

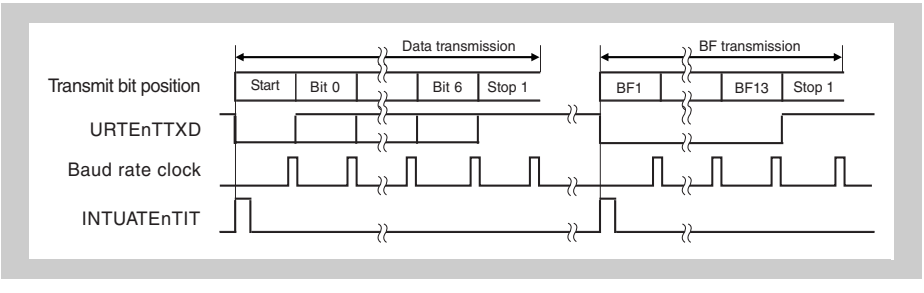


Figure 20-2 Transmission interrupt request timing for URTEnCTL1.URTEnSLIT = 0

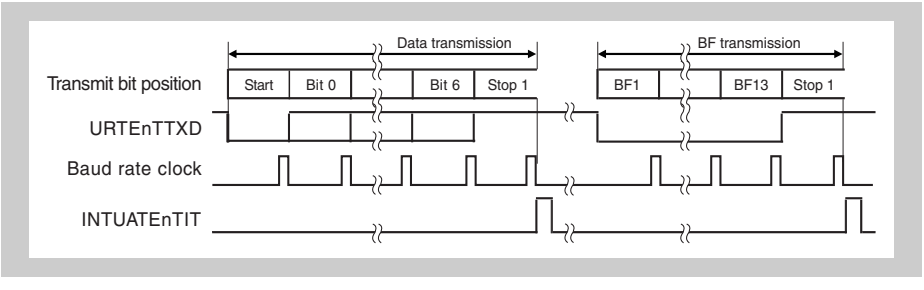


Figure 20-3 Transmission interrupt request timing for URTEnCTL1.URTEnSLIT = 1

20.5.2 Reception interrupt request INTUAEnTIR

A reception interrupt request is generated when the first bit of the stop bit is sampled.

In case of erroneous reception, the status interrupt INTUAEnTIS is generated instead of INTUAEnTIR.

The reception interrupt request INTUAEnTIR is not generated in the reception disabled status.

The following diagrams show the timing of the reception interrupt request during data/BF reception.

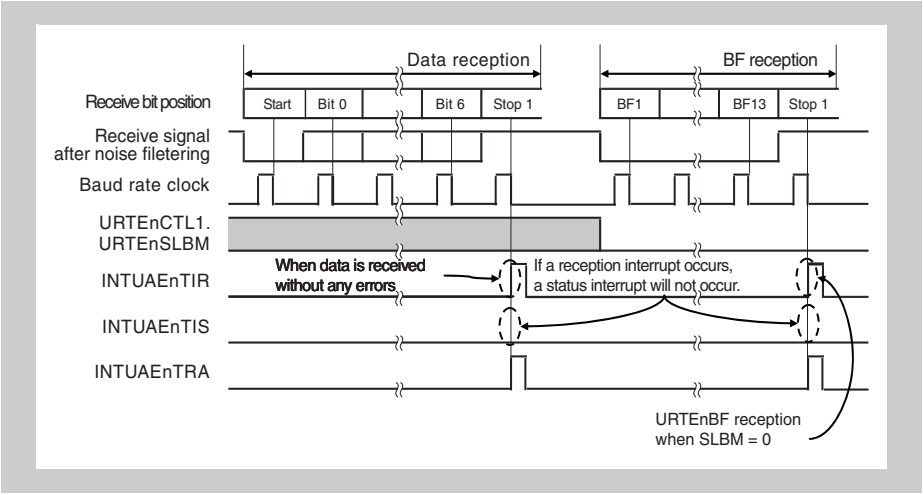


Figure 20-4 Reception interrupt request timing

20.5.3 Status interrupt request INTUAEnTIS

A status interrupt request is generated if an error condition occurred during reception or transmission, as reflected in the status register 1 URTEnSTR1.

If BF reception is enabled (URTEnCTL1.URTEnSLBM = 1) during LIN communication, the status interrupt request signal is generated when a consecutive low level (BF) of 11 bits or more is received.

20.5.4 Receive/status interrupt request INTUAEnTRA

The reception/status interrupt request is asserted, if a reception or status interrupt request is generated. That means:

INTUAEnTRA = INTUAEnTIR or INTUAEnTIS

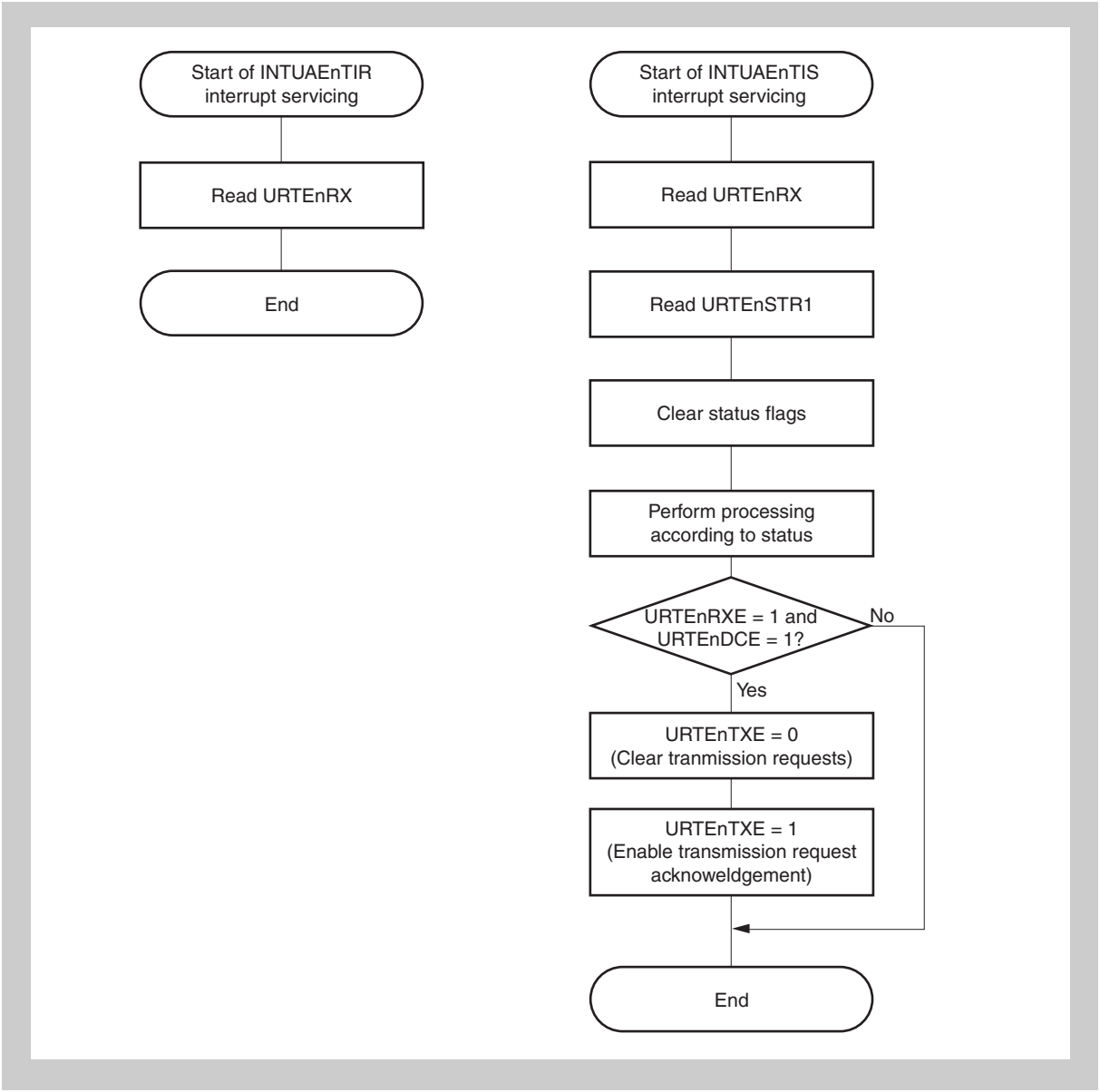


Figure 20-5 Processing flow after interrupt generation

20.6 Operation

20.6.1 Data formats

Full-duplex serial data reception and transmission is performed.

As shown in the figures below, one data frame of transmit/receive data consists of a start bit, character bits, parity bit, and stop bit(s).

Several properties of a transmit/receive data frame can be specified by control bits of the URTEnCTL1 register:

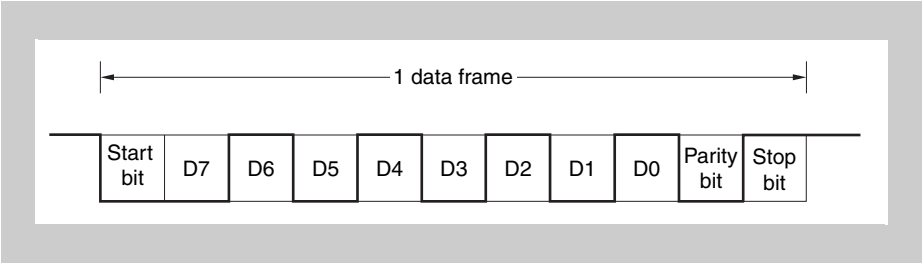
Table 20-18 Data format specification

Item	Options	Control bits
Start bit	1 bit	fixed
Character bits	7 bits / 8 bits	URTEnCTL1.URTEnCLG
Parity	Even parity/odd parity/ 0 parity/no parity	URTEnCTL1.URTEnSLP[1:0]
Stop bit	1 bit / 2 bits	URTEnCTL1.URTEnSLG
Data order	MSB first / LSB first	URTEnCTL1.URTEnSLD
Tx data level	inverted / not inverted	URTEnCTL1.URTEnTDL
Rx data level	inverted / not inverted	URTEnCTL1.URTEnRDL

(1) UARTEn transmit/receive data format

(a) 8-bit data length, LSB first, even parity, 1 stop bit, transfer data: 55_H

(b) 8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55_H



(c) 8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55_H, URTEnTTXD inversion

(d) 7-bit data length, LSB first, odd parity, 2 stop bits, transfer data: 36_H

(e) 8-bit data length, LSB first, no parity, 1 stop bit, transfer data: 87_H

20.6.2 BF transmission/reception format

The UARTEn has an BF (Break Field) transmission/reception control function to enable use of the LIN function.

About LIN

LIN stands for Local Interconnect Network and is a low-speed (1 to 20 kbps) serial communication protocol intended to aid the cost reduction of an automotive network.

LIN communication is single-master communication, and up to 15 slaves can be connected to one master.

The LIN slaves are used to control the switches, actuators, and sensors, and these are connected to the LIN master via the LIN network.

Normally, the LIN master is connected to a network such as CAN (Controller Area Network).

In addition, the LIN bus uses a single-wire method and is connected to the nodes via a transceiver that complies with ISO9141.

In the LIN protocol, the master transmits a frame with baud rate information and the slave receives it and corrects the baud rate error. Therefore, communication is possible when the baud rate error in the slave is ±14% or less.

Figure 20-6 "LIN transmission outline" and *Figure 20-7 "LIN reception outline"* outline the transmission and reception manipulations of LIN.

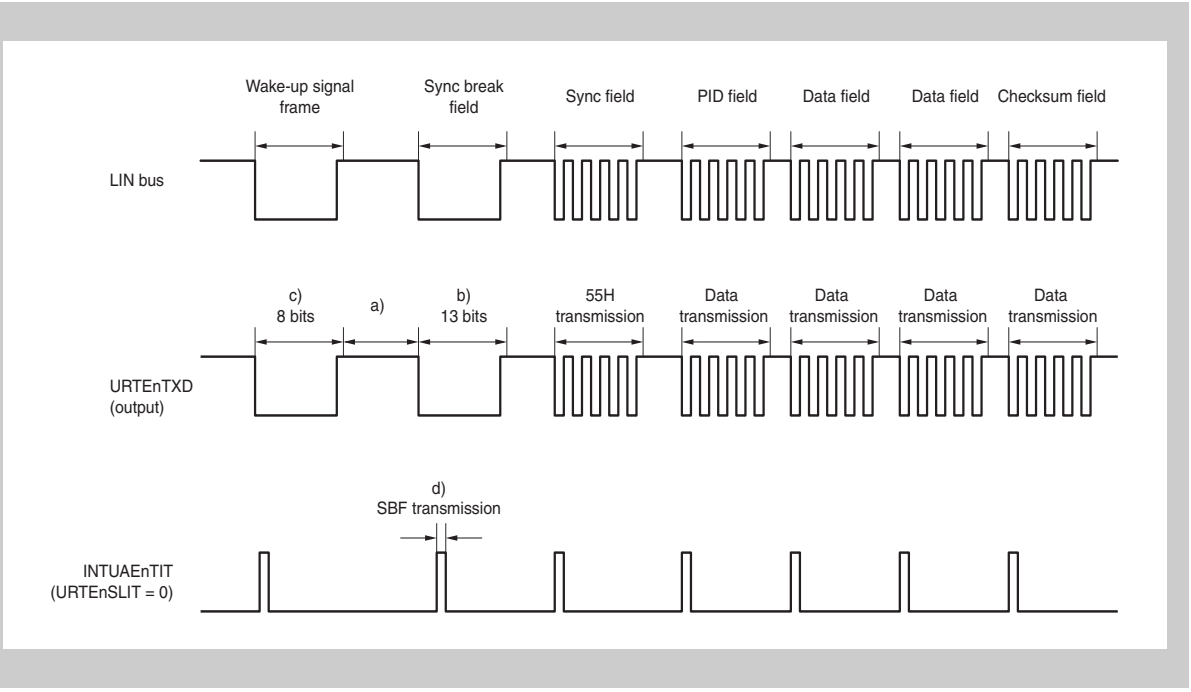


Figure 20-6 LIN transmission outline

a) The interval between fields is controlled by software.
b) BF output is performed by hardware. The output width is the bit length set by URTEnCTL1.URTEnBLG[2:0].
If even finer output width adjustments are required, such adjustments can be performed using URTEnCTL2.URTEnBRS[11:0].
c) 80_H transfer in the 8-bit mode is substituted for the wakeup signal frame.
d) A transmission enable interrupt INTUAEnTIT is generated at the start of each transmission. INTUAEnTIT is also generated at the start of each BF transmission.

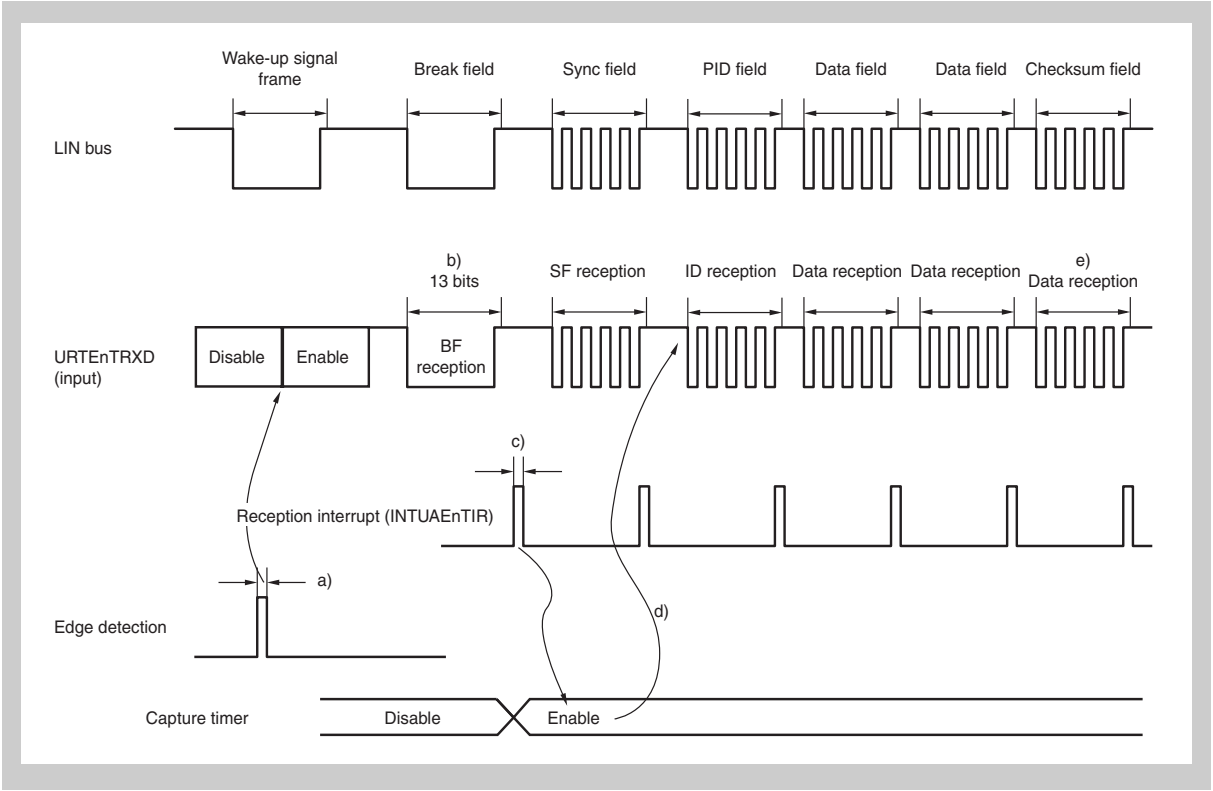


Figure 20-7 LIN reception outline

a) The wakeup signal sent by the pin edge detector enables UARTEn, and sets the BF reception mode.

b) BF reception is judged to end normally when a BF of 11 or more bits is received. An interrupt is generated as shown in the table below, according to the setting of the BF reception mode selection bit URTEnCTL1.URTEnSLBM and the value of the URTEnSTR0.URTEnSSBR bit.

URTEnSLBM	URTEnSSBR	Interrupts
1	x	INTUAEnTIS
0	1	INTUAEnTIR
0	0	A framing error has occurred, so and INTUAEnTIS is generated.

c) When BF reception ends normally, an interrupt is generated as follows according to the setting of the BF reception mode selection bit URTEnCTL1.URTEnSLBM:
- When URTEnDTL1.URTEnSLBM is 0, the reception interrupt INTUAEnTIR is generated.
- When URTEnDTL1.URTEnSLBM is 1, the status interrupt INTUAEnTIS is generated and the BF reception success flag URTEnSTR1.URTEnBSF is set.

If the BF reception trigger bit URTEnTRG.URTEnBRT is 1, error detection for overrun, parity, and framing errors is not performed during BF reception. Also, data transfer from the receive shift register to the receive data register URTEnRX is not performed. URTEnRX holds the previous value at this time.

d) In order to adjust the baud rate clock properly, the URTEnTRXD signal must be connected to the timer capture input. The transfer rate and the baud rate error can be calculated by measuring the time between URTEnTRXD edges, and the baud rate can be adjusted by specifying a value for the baud rate setting bits URTEnCTL2.URTEnBRS[11:0].

e) A checksum field is identified by software. When a checksum field is received, UARTEn is initialized and set to the BF reception mode by software. But if URTEnCTL1.URTEnSLBM is 1 at this time, UARTEn automatically starts BF reception without entering the BF reception mode.

20.6.3 BF transmission

When the URTEnCTL0 bits URTEnPW = URTEnTXE = 1, the transmission enabled status is entered, and BF transmission is started by setting the BF transmission trigger URTEnTRG.URTEnBTT = 1.

Thereafter, URTEnSTR0.URTEnSSBT is set to “1” and a low level width of 13 to 20 bits, as specified by URTEnCTL1.URTEnBLG[2:0], is output.
A transmission interrupt INTUAEnTIT) is generated upon BF

- transmission start, if URTEnCTL1.URTEnSLIT = 0
- transmission end, if URTEnCTL1.URTEnSLIT = 1.

Following the end of BF transmission, URTEnSTR0.URTEnSSBT is automatically cleared. Thereafter, the UARTEn transmission mode is restored.

Transmission is suspended until the data to transmit next is written to the URTEnTX register and URTEnSTR0.URTEnSST is set, or until the BF transmission trigger URTEnTRG.URTEnBTT is set and URTEnSTR0.URTEnSSBT changes to 1.

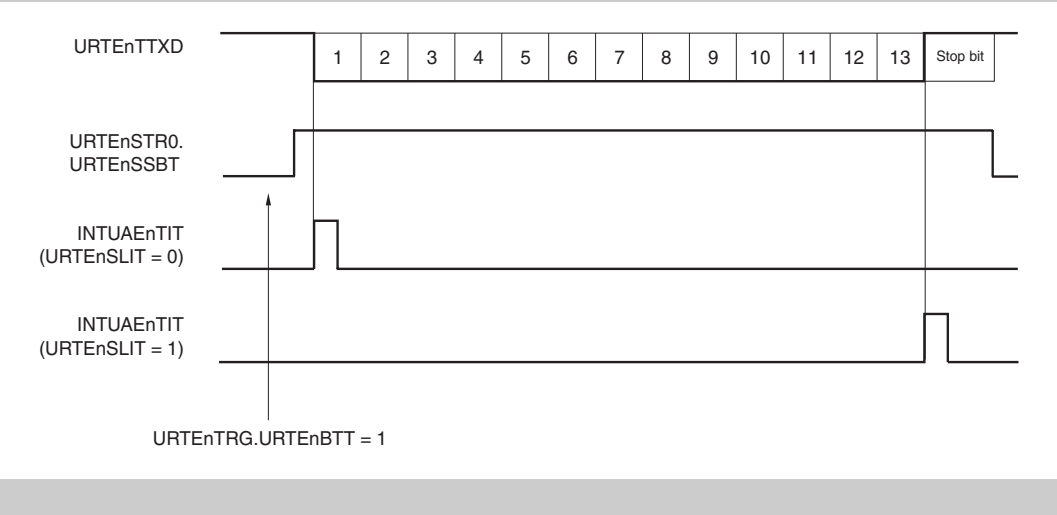
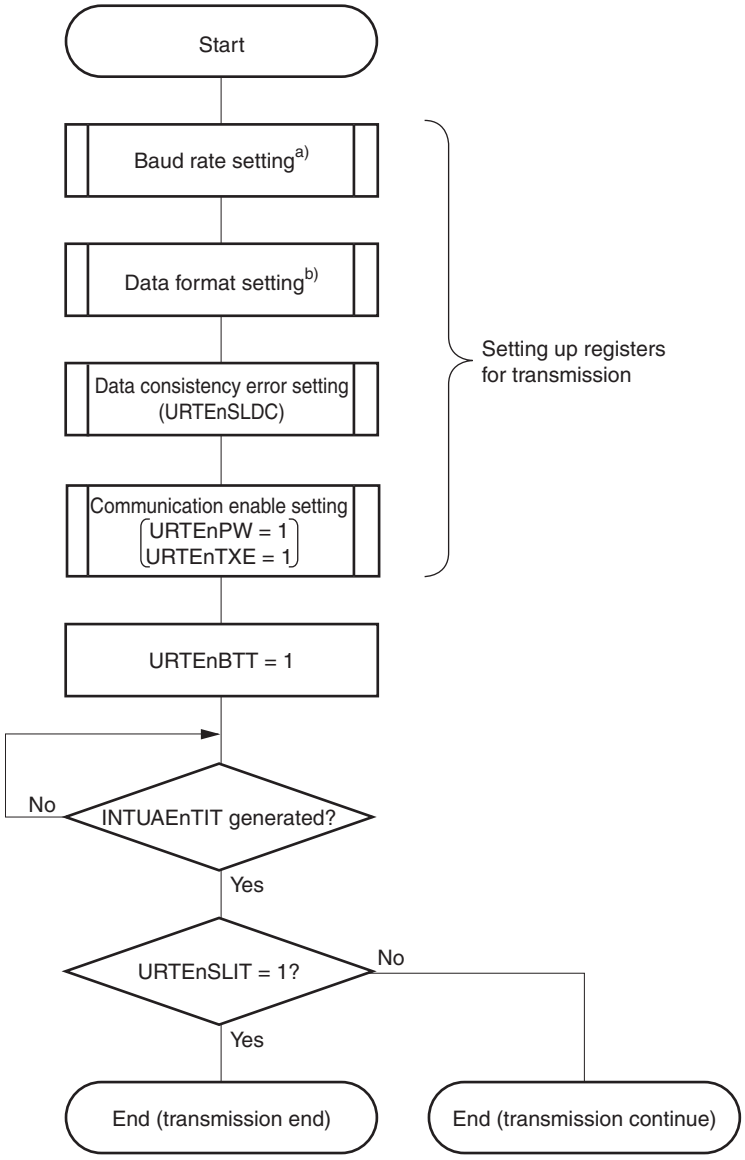


Figure 20-8 BF transmission



a) Set by using the URTEnPRS[2:0] and URTEnBRS[11:0] bits.

b) Set by using the URTEnBLG[2:0], URTEnCLG, URTEnSLP[1,0], URTEnTDL, URTEnSLG, URTEnSLD, and URTEnSLIT bits.

Figure 20-9 Flowchart of BF transmission

20.6.4 BF reception

The reception enabled status is achieved by setting the URTEnCTL0.URTEnPW bit to 1 and then setting the URTEnCTL0.URTEnRXE bit to 1.

The BF reception wait status is set by setting the BF reception trigger URTEnTRG.URTEnBRT = 1.

In the BF reception wait status, similarly to the UARTEn reception wait status, the URTEnTRXD pin is monitored and start bit detection is performed.

Following detection of the low level, reception is started and the internal counter counts up according to the set baud rate.

When a high level is received and if the BF width is 11 or more bits, while the BF receiving mode selection bit

- URTEnCTL1.URTEnSLBM = 0,
  the reception interrupt INTUAEnTIR is generated.
- URTEnCTL1.URTEnSLBM = 1,
  the status interrupt INTUAEnTIS is generated and BF reception success flag URTEnSTR1.URTEnBSF is set at the same time.

The URTEnSTR0.URTEnSSBR bit is automatically cleared and BF reception ends.

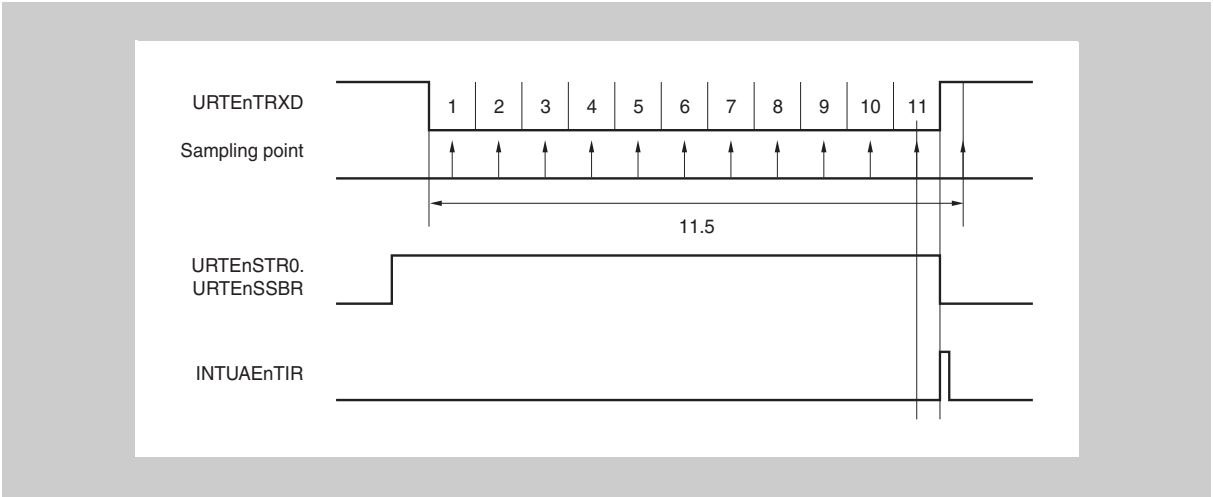


Figure 20-10 Normal BF reception (stop bit after more than 10.5 “L” bits)

Error detection for the URTEnSTR1 error flags URTEnOVE, URTEnPE, and URTEnFE is suppressed and UARTEn communication error detection processing is not performed.
Moreover, the erroneous data is not stored in URTEnRX, but the initial value FFH is held.

If the BF width is 10 or fewer bits, reception is terminated as error processing without generating an interrupt, and the BF reception mode is returned to. URTEnSTR0.URTEnSSBR is not cleared at this time.

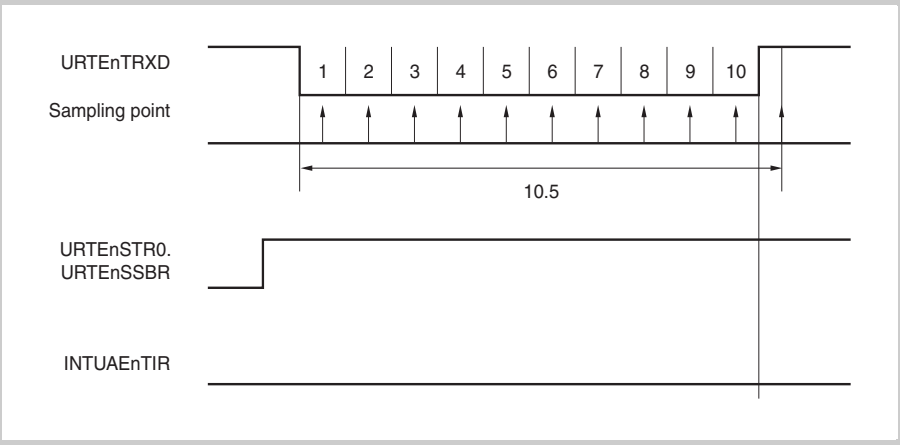


Figure 20-11 BF reception error (stop bit within 10.5 “L” bits)

The BF mode can be selected between a single BF receive mode and an any time BF receive mode in by URTEnCTL1.URTEnSLBM. The status of a successful reception of the BF is indicated by URTEnSTR1.URTEnBSF.

Note URTEnSTR0.URTEnSSBR is set to “1” when

- URTEnTRG.URTEnBRT is set to “1”, or
- the error is cleared by normal BF reception.

20.6.5 Transmission data consistency check

The UARTEn incorporates a data consistency check function to detect a mismatch between the transmit data written to transmit register URTEnTX and the data on the bus when the device operates in master mode.

Data consistency check is enabled by URTEnCTL0.URTEnSLDC = 1.

The data consistency is checked by comparing the transmit data in the transmit register URTEnTX and the receive data in the receive register URTEnRX. In case of a mismatch the data consistency error flag URTEnSTR1.URTEnDCE is set and a status interrupt request INTUAEnTIS occurs.

The consistency check of data is not done in reception mode.

The consistency check of the send data and the input data terminal level is done even if the reception is disabled during sending. In that case also the reception completion interrupt request signal INTUAEnTIR, the URTEnSTR1 status bits URTEnBSF, URTEnFE, URTEnOVE and the status interrupt request signal INTUAEnTIS will not be generated as well. Receive data does not need to be read.

See *(6) "URTEnSTR1 - UARTEn status register 1" on page 1389* for details.

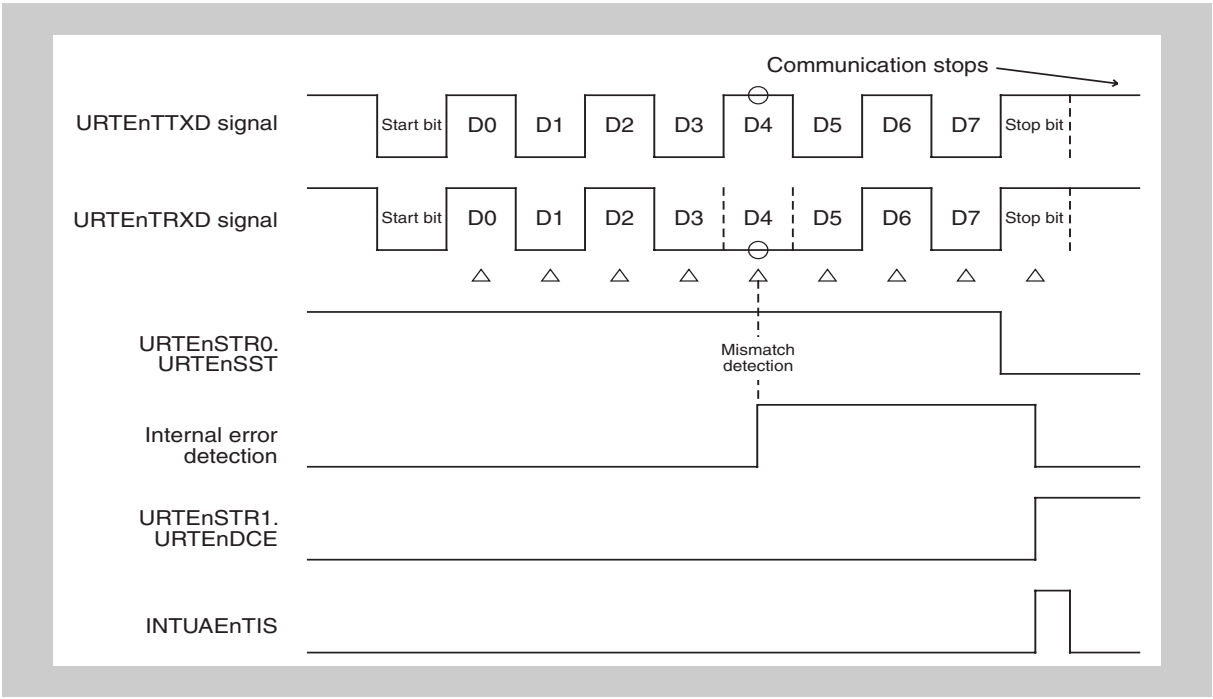


Figure 20-12 Timing example of data consistency error (no BF reception active, i.e. URTEnSTR0.URTEnSSBR = 0)

20.6.6 UARTEn transmission

Transmission start

Set the transmission enabled status by performing the following procedures.

- Specify the baud rate by the UARTEn control register 2 URTEnCTL2.
- Specify the transmit parity, data character length, stop bit length, transmit data order, transmission interrupt request timing and output logic level by the UARTEn control register 1 URTEnCTL1.
- Enable UARTEn operation and transmission by URTEnCTL0.URTEnPW = URTEnCTL0.URTEnTXE = 1)

Write of the transmit data to the transmission buffer register URTEnTX starts transmission. The data which is saved in the URTEnTX register is transferred to the transmit shift register URTEnTXS. Then, the start, parity and stop bits are added and the data frame is output serially via URTEnTTXD.

Transmission stop

When URTEnCTL0.URTEnPW or URTEnCTL0.URTEnTXE is set to 0, transmission operations are stopped immediately, even during transmission processing.

Concurrent BF and data transmission

When a BF transmit request and a data transmit request have both been set, BF transmission takes priority.

Data consistency check

If a data consistency error is detected, the subsequent data is not transmitted until URTEnCLDC is set to 1, URTEnPW is set to 0, or URTEnTXE is set to 0.

INTUAEnTIT timing

The time required to generate the transmission interrupt INTUAEnTIT depends on the setting of the URTEnCTL1.URTEnSLIT bit:

- URTEnCTL1.URTEnSLIT = 0
  INTUAEnTIT is generated at the start of transmission, i.e. when the data from the data register URTEnTX is transferred to the transmit shift register and transmission is started.
- URTEnCTL1.URTEnSLIT = 1
  INTUAEnTIT is generated when the entire data transmission process is completed, i.e. when the last bit of the data frame has been transmitted.

Once INTUAEnTIT is generated, the next data can be written to URTEnTX.

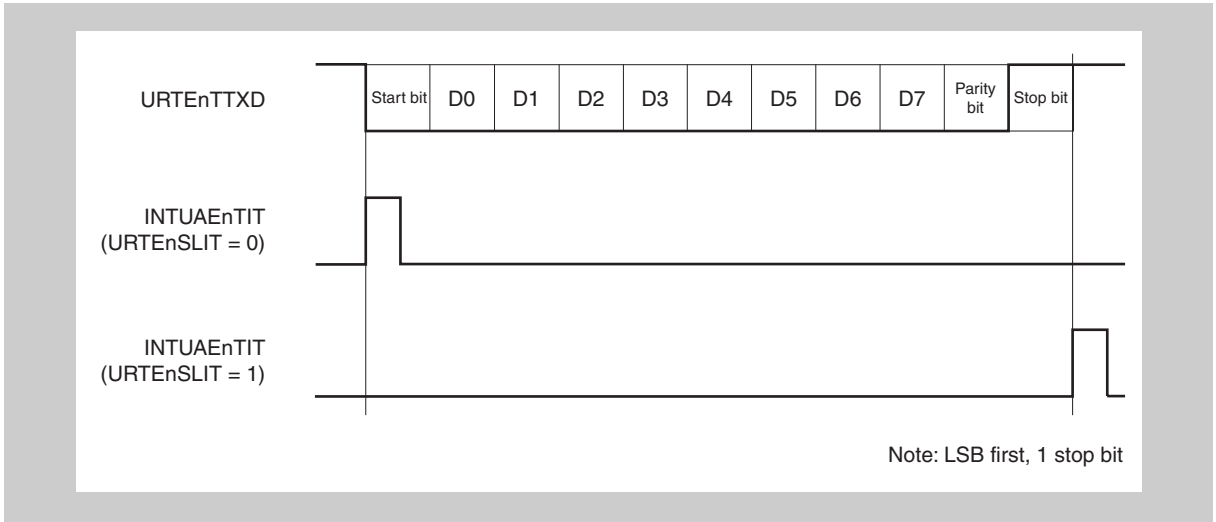
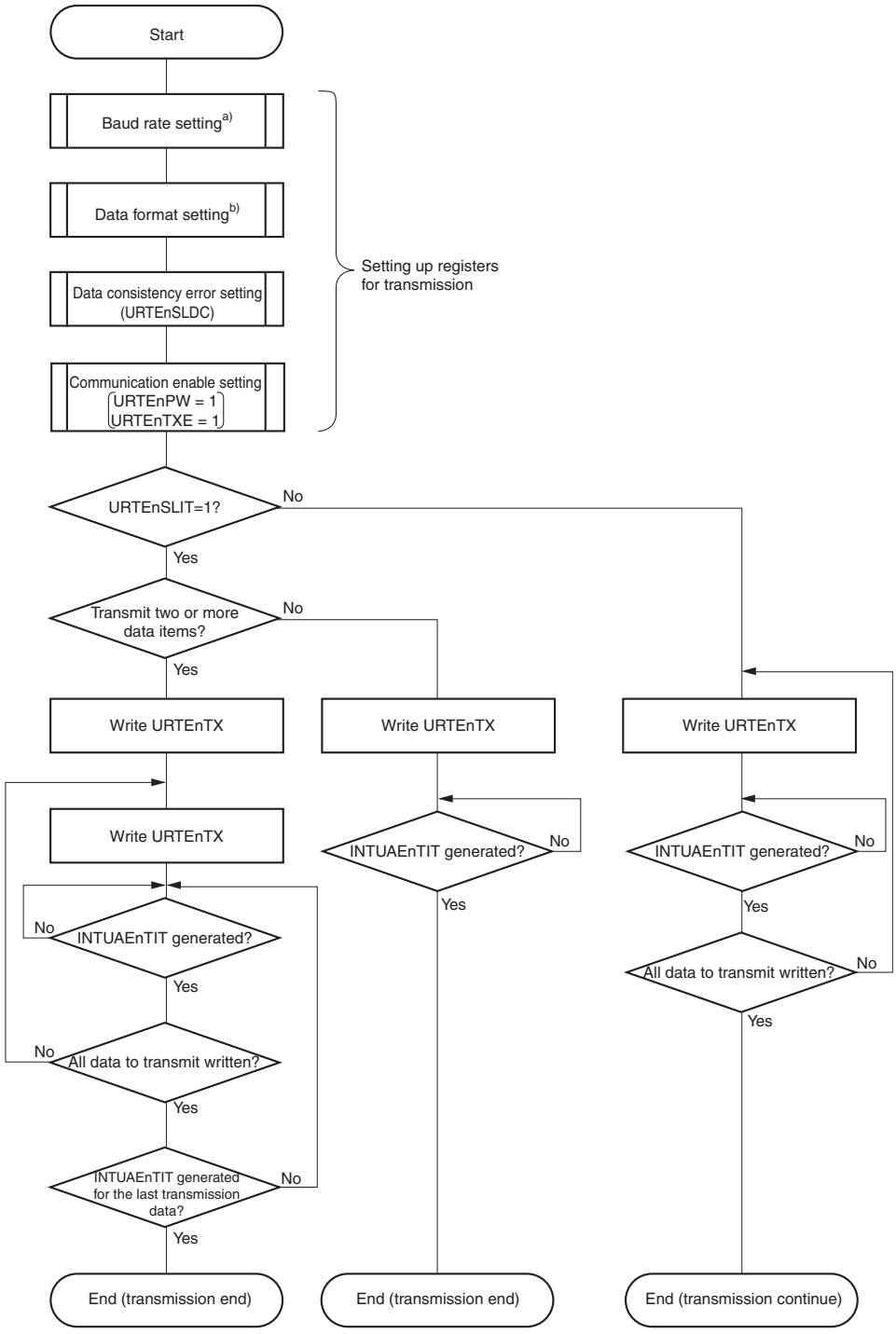


Figure 20-13 Transmission interrupt timing



a) Set by using the URTEnPRS[2:0] and URTEnBRS[11:0] bits.

b) Set by using the URTEnCLG, URTEnSLP[1,0], URTEnTDL, URTEnSLG, URTEnSLD, and URTEnSLIT bits.

Figure 20-14 Flowchart of data transmission

20.6.7 Continuous transmission procedure

Continuous transmission is achieved by writing the next transmit data to the transmit data register, while shifting out of the previous data from the transmission shift register URTEnTXS is ongoing.

Note In order to maintain correct write timing, the transmission interrupt INTUAEnTIT must be generated at the start of each transmission (URTEnCTL1.URTEnSLIT = 0).

Caution If the value is written to the URTEnTX register before the INTUAEnTIT is generated, the transmit data set before is overwritten by the new transmit data.

To initialize the transmission unit, confirm that no transmission is ongoing (URTEnSTR0 bits URTEnSSBT = URTEnSST = 0). If the initialization is performed during an ongoing transmission, the transmission is aborted.

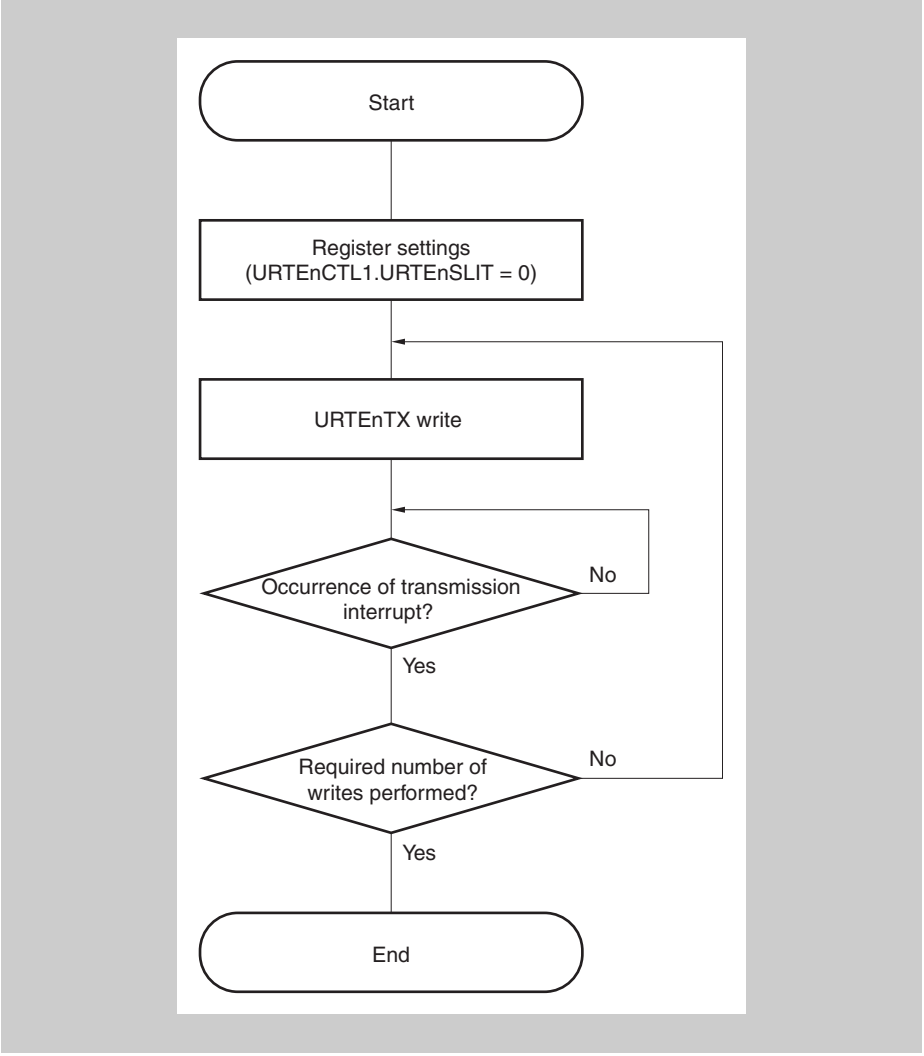


Figure 20-15 Continuous transmission processing flow

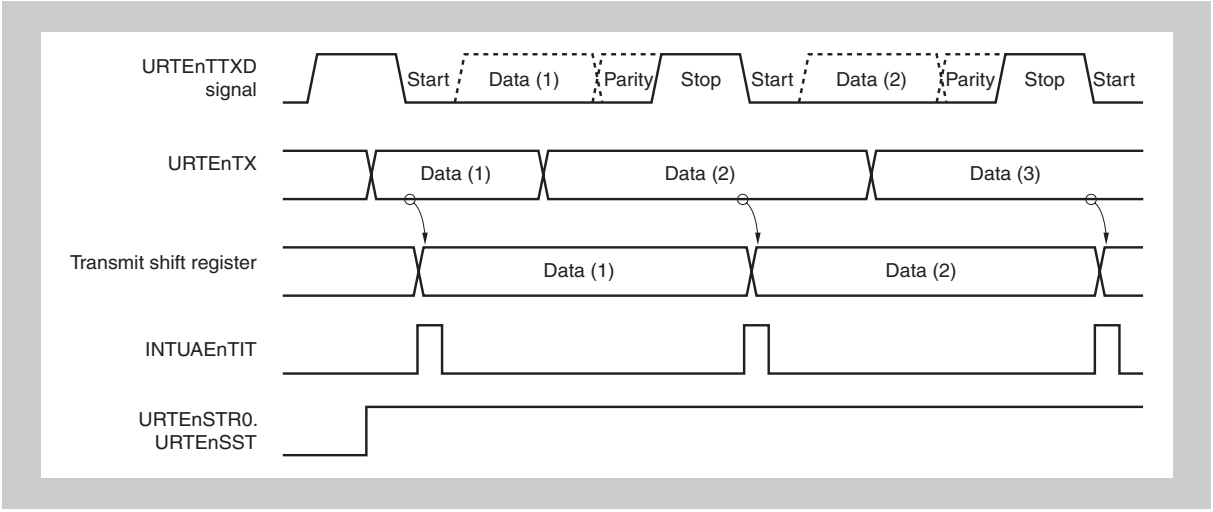


Figure 20-16 Continuous transmission operation timing - transmission start

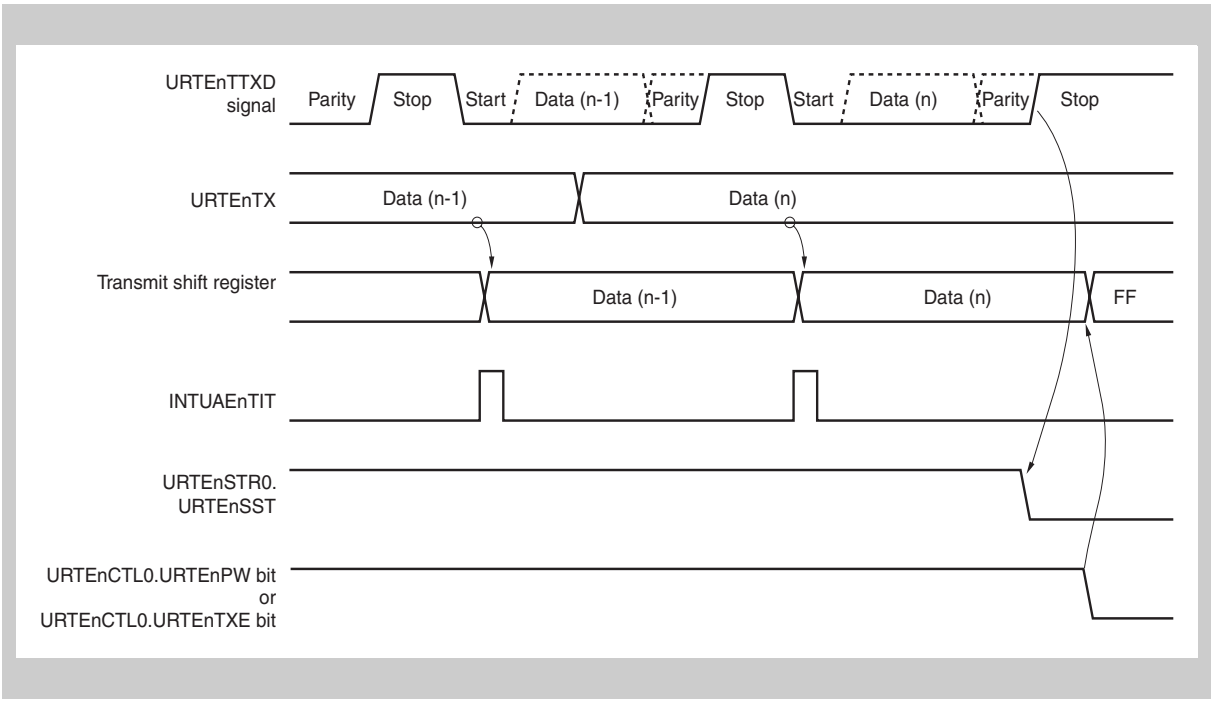


Figure 20-17 Continuous transmission operation timing - transmission end

20.6.8 UARTEn reception

Reception start

Set the reception enabled status by using the following procedure:

- Specify the baud rate by URTEnCTL2.
- Specify the receive parity, data character length, stop bit length, receive data order and output logic level by URTEnCTL1.
- Enable UARTEn operation and reception by setting URTEnCTL0.URTEnPW and URTEnCTL0.URTEnRXE.

When the falling edge of the URTEnTRXD pin input level is detected, data sampling of the URTEnTRXD input is started. The start bit is recognized if the URTEnTRXD pin is low level after the time of a half bit is passed after the detection of the falling edge (shown in the figure below). After a start bit has been recognized, the receive operation starts, and serial data is stored in the receive shift register according to the set baud rate. When the reception interrupt INTUAEnTIR is asserted upon reception of the stop bit, the data stored in the receive shift register is written to the receive data register URTEnRX.

Reception stop

When URTEnCTL0.URTEnPW or URTEnCTL0.URTEnRXE is set to 0, reception operations are stopped immediately, even during reception processing.

Reception errors

If an overrun error occurs (URTEnSTR1.URTEnOVE = 1), the receive data at this time is not transferred to the URTEnRX register and is discarded.
Even if a parity error (URTEnSTR1.URTEnPE = 1) or a framing error (URTEnSTR1.URTEnFE = 1) occurs during reception, reception continues until the reception position of the first stop bit, and the reception data is transferred to the URTEnRX.
When a reception error occurs, the status interrupt INTUAEnTIS –and therefore also the receive/status interrupt INTUAEnTRA– is generated, but not the reception interrupt INTUAEnTIR.

To change the receive data order, parity, data character length, or the stop bit length, clear the power bit (URTEnCTL0.URTEnPW = 0) first.

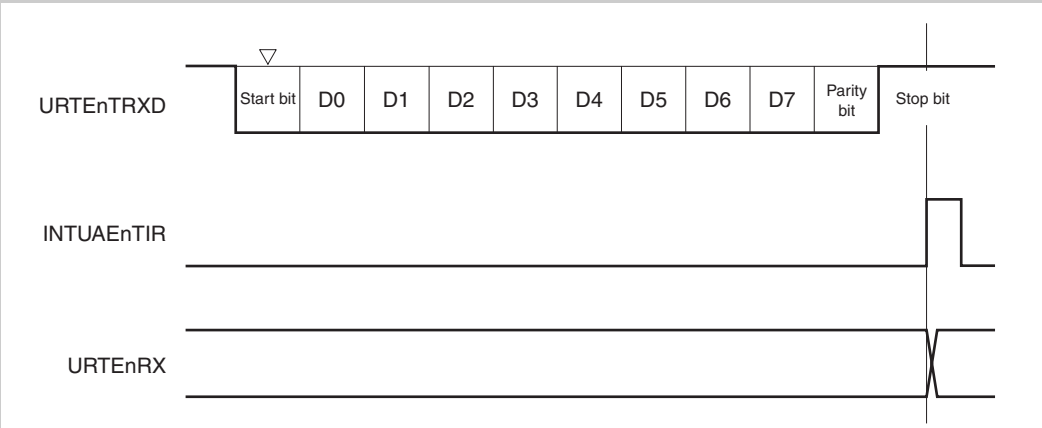


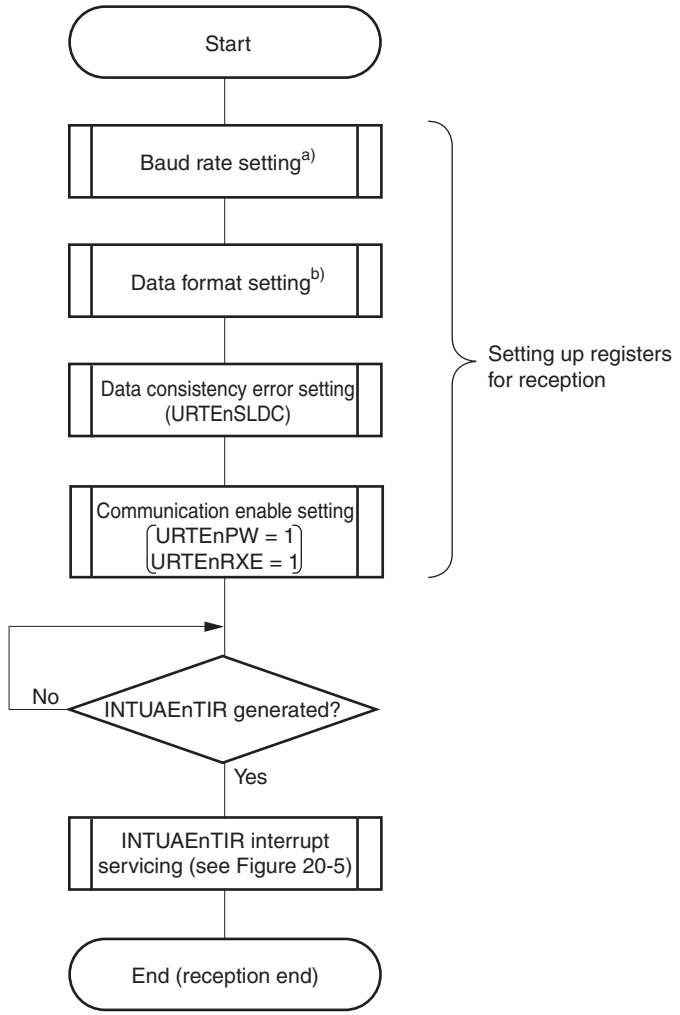
Figure 20-18 UARTEn reception

Cautions

1. Be sure to read the URTEnRX register even when a reception error occurs. If the URTEnRX register is not read, an overrun error occurs during reception of the next data.
2. The operation during reception is performed assuming that there is only one stop bit. A second stop bit is ignored.
3. When reception is completed, read the URTEnRX register after the reception interrupt INTUAEnTIR has been generated, and clear the URTEnCTL0.URTEnPW or URTEnCTL0.URTEnRXE bit to 0.
   If the URTEnCTL0.URTEnPW or URTEnCTL0.URTEnRXE bit is cleared to 0 before the INTUAEnTIR is generated, the read value of the URTEnRX register cannot be guaranteed.
4. If receive completion processing (INTUAEnTIR interrupt generation) and the URTEnCTL0.URTEnPW bit = 0 or URTEnCTL0.URTEnRXE bit = 0 conflict, INTUAEnTIR may be generated in spite of these being no data stored in the URTEnRX register.

Notes

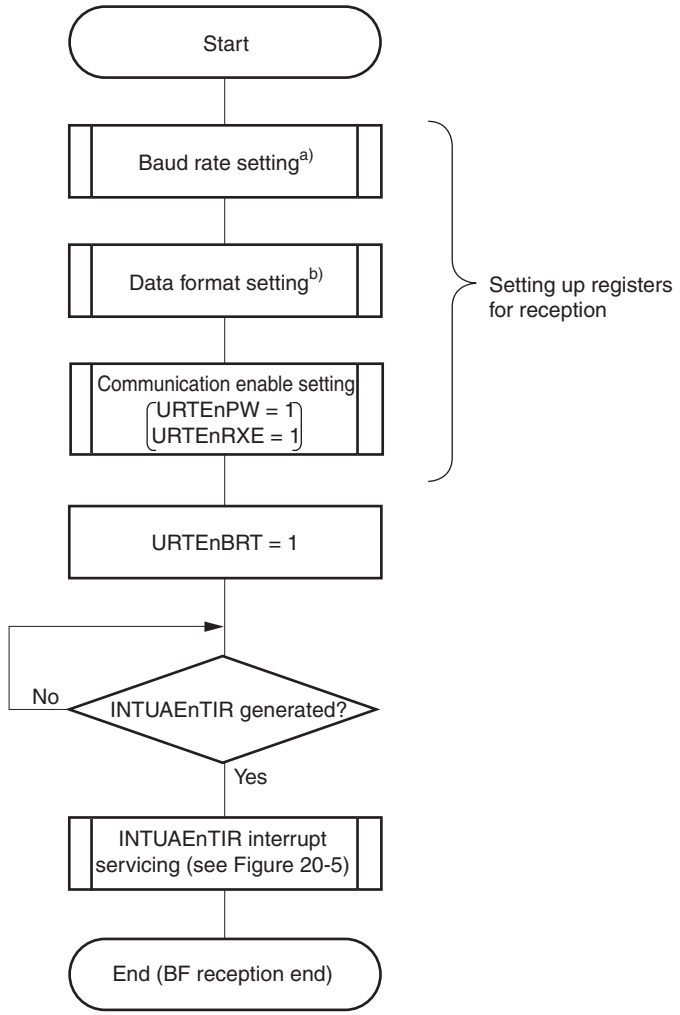
1. If low level is always input to the URTEnTRXD pin, it is not judged as the start bit.
2. In continuous reception, immediately after the stop bit is detected at the first reception bit (when the reception interrupt is generated), the next start bit can be detected.



a) Set by using the URTEnPRS[2:0] and URTEnBRS[11:0] bits.

b) Set by using the URTEnSLBM, URTEnCLG, URTEnSLP[1,0], URTEnRDL, and URTEnSLD bits.

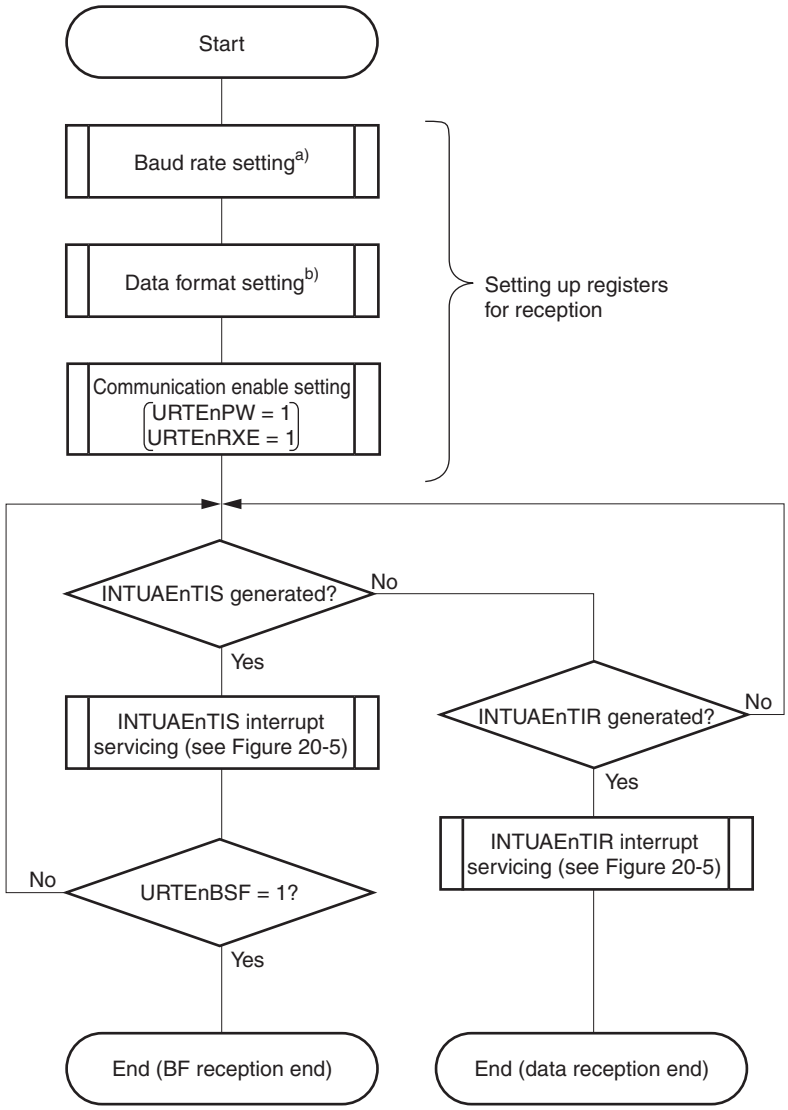
Figure 20-19 Flowchart of data reception when URTEnSLBM = 0, URTEnSSBR = 0



a) Set by using the URTEnPRS[2:0] and URTEnBRS[11:0] bits.

b) Set by using the URTEnSLBM, URTEnCLG, URTEnSLP[1,0], URTEnRDL, and URTEnSLD bits.

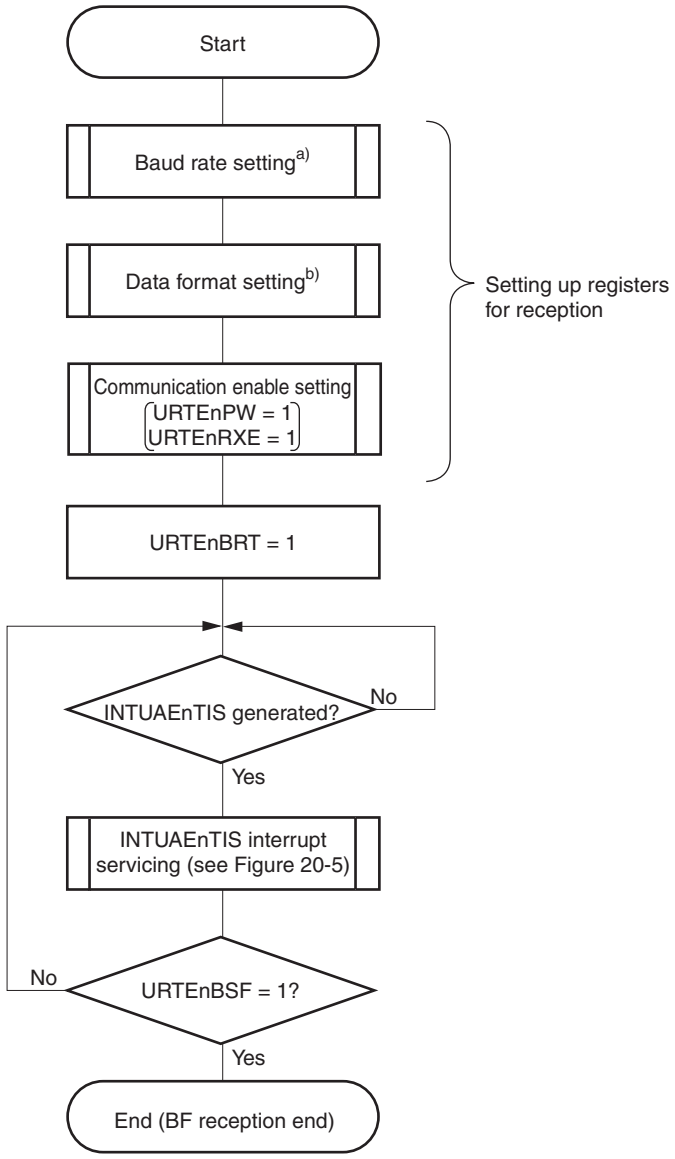
Figure 20-20 Flowchart of data reception when URTEnSLBM = 0, URTEnSSBR = 1



a) Set by using the URTEnPRS[2:0] and URTEnBRS[11:0] bits.

b) Set by using the URTEnSLBM, URTEnCLG, URTEnSLP[1,0], URTEnRDL, and URTEnSLD bits.

Figure 20-21 Flowchart of data reception when URTEnSLBM = 0, URTEnSSBR = 0



a) Set by using the URTEnPRS[2:0] and URTEnBRS[11:0] bits.

b) Set by using the URTEnSLBM, URTEnCLG, URTEnSLP[1,0], URTEnRDL, and URTEnSLD bits.

Figure 20-22 Flowchart of data reception when URTEnSLBM = 1, URTEnSSBR = 1

20.6.9 Reception errors

Errors during a receive operation are of three types: parity errors, framing errors, and overrun errors. Data reception result error flags are set in the URTEnSTR1 register and a status interrupt request signal INTUAEnTIS is generated when an error occurs.

It is possible to ascertain which error occurred during reception by reading the contents of the URTEnSTR1 register.

Clear a reception error flag by writing 1 to its associated bit in the status clear register URTEnSTC.

Table 20-19 Reception error causes

Error flag in URTEnSTR1	Reception error	Cause
URTEnPE	Parity error	Received parity bit does not match the setting
URTEnFE	Framing error	Stop bit not detected
URTEnOVE	Overrun error	Reception of next data completed before data was read from receive buffer

Note Even in case of a parity or framing error, data is transferred from the receive shift register to the receive data register URTEnRX. Consequently the data from URTEnRX must be read. Otherwise an overrun error URTEnSTR1.URTEnOVE will occur at reception of the next data.

In case of an overrun error, the receive shift register data is not transferred to URTEnRX, thus the previous data is not overwritten.

20.6.10 Parity types and operations

Caution When using the LIN function, fix the URTEnCTL1.URTEnSLP[1:0] to 00_B.

The parity bit is used to detect bit errors in the communication data. Normally the same parity is used on the transmission side and the reception side.

In the case of even parity and odd parity, it is possible to detect odd-count bit errors. In the case of 0 parity and no parity, errors cannot be detected.

(1) Even parity

- During transmission
  The number of bits whose value is “1” among the transmit data, including the parity bit, is controlled so as to be an even number. The parity bit values are as follows:
  - Odd number of bits whose value is “1” among transmit data:1
  - Even number of bits whose value is “1” among transmit data:0
- During reception
  The number of bits whose value is “1” among the reception data, including the parity bit, is counted, and if it is an odd number, a parity error is output.

(2) Odd parity

- During transmission
  Opposite to even parity, the number of bits whose value is “1” among the transmit data, including the parity bit, is controlled so that it is an odd number. The parity bit values are as follows.
  - Odd number of bits whose value is “1” among transmit data: 0
  - Even number of bits whose value is “1” among transmit data: 1
- During reception
  The number of bits whose value is “1” among the receive data, including the parity bit, is counted, and if it is an even number, a parity error is output.

(3) 0 parity

During transmission, the parity bit is always made 0, regardless of the transmit data.

During reception, parity bit check is not performed. Therefore, no parity error occurs, regardless of whether the parity bit is 0 or 1.

(4) No parity

No parity bit is added to the transmit data.

Reception is performed assuming that there is no parity bit. No parity error occurs because there is no parity bit.

20.6.11 Digital receive data noise filter

The receive data signal input URTEnTRXD is equipped with a digital noise filter to eliminate noise and spikes.

This filter samples the URTEnTRXD pin using the prescaler output clock PRSCLK.

When the same sampling value is read twice, the URTEnTRXD signal is validated as the input data.
Therefore, data not exceeding the width of 2 prescaler output clocks is judged to be noise and thus eliminated.

The noise filter causes a delay of 4 prescaler output clock PRSCLK cycles when capturing the serial data URTEnTRXD, until it is forwarded as valid.

20.7 Baud Rate Generator

The transmission and reception baud rate BRCLK are derived from the APB bus clock PCLK by use of a prescaler and a baud rate generator, as shown in the figure below.

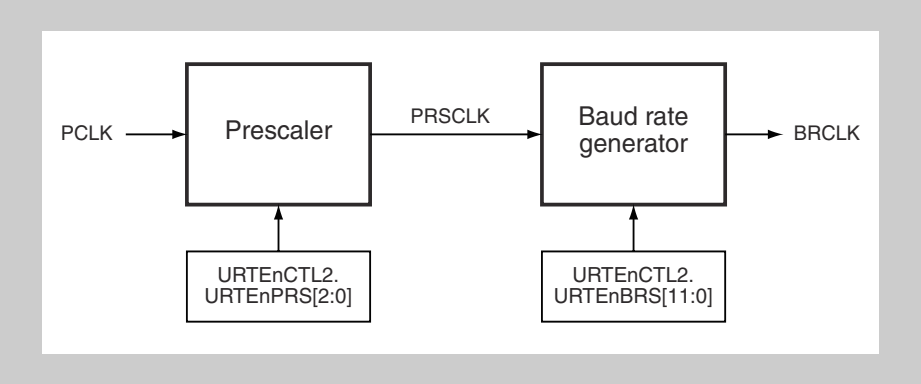


Figure 20-23 Configuration of baud rate generator

The prescaler output clock PRSCLK is a fraction of PCLK, the divisor is set up the value URTEnCTL2.URTEnPRS[2:0]:

$$PRSCLK = PCLK / 2^{URTEnPRS[2:0]}$$

PRSCLK is further divided by the baud rate generator by a value, determined by URTEnCTL2.URTEnBRS[11:0].
The baud rate generator distinguishes between the baud rate for data frames and BF receptions, as listed in the table below. The BF reception clock is the double of the baud rate clock BRCLK.

Table 20-20 Baud rate generator clocks output

URTEnCTL2. URTEnBRS[11:0]	Transmit/receive BRCLK	BF receive clock
000_H	PCLK / (2 x 4)	PCLK / 4
001_H		
002_H		
003_H		
004_H		
005_H	PCLK / (2 x 5)	PCLK / 5
...	PCLK / (2 x URTEnBRS[11:0])	PCLK / URTEnBRS[11:0]
FFE_H	PCLK / (2 x 4094)	PCLK / 4094
FFF_H	PCLK / (2 x 4095)	PCLK / 4095

Chapter 21 LIN Master Controller (LMA)

This chapter describes the LIN master controller (LMA).

The first section describes all properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names. The subsequent sections describe the features that apply to all implementations.

21.1 V850E2/Sx4-H LMAn Features

Instances

This microcontroller has following number of instances of LMA:

Table 21-1 Instances of LMAn

<R>

LMA	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of instances	4	5	5
Name	LMA0 to LMA3	LMA0 to LMA3, LMA10	LMA0 to LMA3, LMA10

LMAn instances index n

Throughout this chapter, the instance of LMA is identified by the index "n" (n = 0 to 11), for example, LMAnCTLL for the LMAn control register L.

Channels

This microcontroller has following number of channels of the LIN master scheduler counter (CNTA):

Table 21-2 Channels of CNTAn

<R>

CNTA	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of channels	2	3	3
Name	CNTA0, CNTA1	CNTA0 to CNTA2	CNTA0 to CNTA2

CNTAm instances index m

Throughout this chapter, the instance of CNTA is identified by the index "m" (n = 0 to 2), for example, CNTAmCTL for the CNTAm control register.

LMAn register addresses

All LMAn register addresses are given as addresses offset from the individual base address <LMAn_base>.
The <LMAn_base> address of each LMAn are listed in the following table:

Table 21-3 LMAn register base addresses <LMAn_base>

LMAn	<LMAn_base> address
LMA0	FF5C 0080_H
LMA1	FF5D 0080_H
LMA2	FF5E 0080_H
LMA3	FF5F 0080_H
LMA10	FF66 0080_H

CNTAm register addresses

All CNTAm register addresses are given as addresses offset from the individual base address <CNTAm_base>.
The <CNTAm_base> address of each CNTAm are listed in the following table:

Table 21-4 CNTAm register base addresses <CNTAm_base>

CNTAm	<CNTAm_base> address
CNTA0	FF5C 4000_H
CNTA1	FF5D 4000_H
CNTA2	FF5E 4000_H

Clock supply

The following clock is supplied to LMA and CNTA:

Table 21-5 LMAn/CNTAm clock supply

LMAn/CNTAm	Clock	Connected to:
LMA0	PCLK	Clock generator CKSCL_112
LMA1	PCLK	Clock generator CKSCL_112
LMA2	PCLK	Clock generator CKSCL_114
LMA3	PCLK	Clock generator CKSCL_114
LMA10	PCLK	Clock generator CKSCL_011
CNTA0	PCLK	Clock generator CKSCL_112
CNTA1	PCLK	Clock generator CKSCL_114
CNTA2	PCLK	Clock generator CKSCL_011

Interrupts and DMA

LMA can generate the following interrupt requests and DMA requests:

Table 21-6 LMAn interrupts and DMA requests

LMAn signals	Function	Connected to:
LMA0:		
INTLMA0TIT	Transmission interrupt	Interrupt controller INTLMA0IT DMA controller trigger 45
INTLMA0TIR	Reception interrupt	Interrupt controller INTLMA0IR DMA controller trigger 44
INTLMA0TIS	Status interrupt	Interrupt controller INTLMA0IS
LMA1:		
INTLMA1TIT	Transmission interrupt	Interrupt controller INTLMA11IT DMA controller trigger 47
INTLMA1TIR	Reception interrupt	Interrupt controller INTLMA11IR DMA controller trigger 46
INTLMA1TIS	Status interrupt	Interrupt controller INTLMA11IS
LMA2:		
INTLMA2TIT	Transmission interrupt	Interrupt controller INTLMA2IT DMA controller trigger 92
INTLMA2TIR	Reception interrupt	Interrupt controller INTLMA2IR DMA controller trigger 94
INTLMA2TIS	Status interrupt	Interrupt controller INTLMA2IS
LMA3:		
INTLMA3TIT	Transmission interrupt	Interrupt controller INTLMA3IT DMA controller trigger 118
INTLMA3TIR	Reception interrupt	Interrupt controller INTLMA3IR DMA controller trigger 117
INTLMA3TIS	Status interrupt	Interrupt controller INTLMA3IS
LMA10:		
INTLMA10TIT	Transmission interrupt	Interrupt controller INTLMA10IT DMA controller trigger 93
INTLMA10TIR	Reception interrupt	Interrupt controller INTLMA10IR DMA controller trigger 95
INTLMA10TIS	Status interrupt	Interrupt controller INTLMA10IS

LMA/CNTA hardware reset

LMA and CNTA, and their registers are initialized by the following reset signal:

Table 21-7 LMAn/CNTAm reset signal

LMAn/CNTAm	Reset signal
LMAn	System reset SYSRES
CNTAm	System reset SYSRES

Internal signals

Each LMAn is connected to an asynchronous serial interfaces E URTEn signals and to one of the three CNTAm channels (m = 0 to 2).
The internal signal connections of LMA are listed in the following table:

Table 21-8 LMAn internal signal connections

LMAn signal	Function	Connected to:
LMA10:		
INTUAEnTIT	Transmission interrupt	URTEn INTUAEnTIT
INTUAEnTIR	Reception interrupt	URTEn INTUAEnTIR
INTUAEnTIS	Status interrupt	URTEn INTUAEnTIS
CNTAnCNT[15:0]	Free-running counter value	CNTA2 CNT2CNT[15:0]

LIN master scheduler counters

The correspondence between CNTAm channels and LMAn instances is shown in the following table:

Table 21-9 CNTAm to LMAn assignment

CNTAm	LMAn
CNTA0	LMA0, LMA1
CNTA1	LMA2, LMA3
CNTA2	LMA10

For details about CNTAm, see *21.2 “LIN Master Scheduler Counters (CNTA)”*.

21.2 LIN Master Scheduler Counters (CNTA)

The LIN master scheduler counter consists of a free-running 16-bit counter. The count clock is derived from the CNTA input clock PCLK, that is divided by a prescaler.

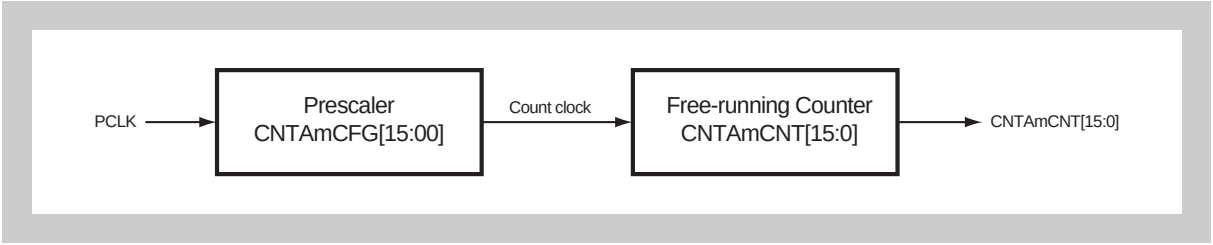


Figure 21-1 LIN master scheduler counter

CNTAm enable Before the LIN master controller enables the scheduler, the counter CNTAm must be enabled by setting CNTAmCTL.CNTAmPW = 1.

Prescaler division The division factor of the prescaler is determined by the value CNTAmCFG.CNTAmPRS[15:00]:

- CNTAmPRS[15:00] = $FFFF_H$: count clock = PCLK / 1
- else: count clock = PCLK / (CNTAmPRS[15:0]+2)

21.2.1 CNTAm registers

The CNTAm is controlled and operated by the following registers:

Table 21-10 CNTAm registers

Register name	Symbol	Address
Control register	CNTAmCTL	<CNTAm_base> + 00_H
Configuration register	CNTAmCFG	<CNTAm_base> + 04_H

<CNTAm_base> The base addresses <CNTAm_base> of the CNTAm are defined in the first section of this chapter under the key word “CNTAm register addresses”.

(1) CNTAmCTL - CNTAm control register

This register enables/disables the CNTAm operation.

Access This register can be read or written in 16-bit units.

Address <CNTAm_base> + 00_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTAm PW	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-11 CNTAmCTL register contents

Bit position	Bit name	Function
15	CNTAm PW	CNTAm operation enable 0: CNTAm disabled 1: CNTAm enabled

(2) CNTAmCFG - CNTAm configuration register

This register sets the division factor for the clock prescaler.

Access This register can be read or written in 16-bit units.

Address <CNTAm_base> + 04_H

Initial Value $FFFF_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTAmPRS[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-12 CNTAmCTL register contents

Bit position	Bit name	Function
15	CNTAm PRS[15:00]	CNTAm prescaler division ratio $FFFF_H$: PCLK/1 0000_H: PCLK/2 0001_H: PCLK/3 0002_H: PCLK/4 ... $FFFE_H$: PCLK/65,536

21.3 Functional Overview

The LMAn module is connected to a UARTEn module. This combination provides a LIN master interface, but can be used also as a buffered UARTEn.

- UART through mode
- UART buffer mode, full-duplex operation
  - 12 byte Tx buffer
  - 12 byte Rx buffer
- LIN master mode
  - automatic checksum generation and check
  - automatic transmission of Break Field (BF), Sync Field (SF), and checksum
  - Scheduler and automatic frame start function

The block diagram shows the environment of the LIN master controller.

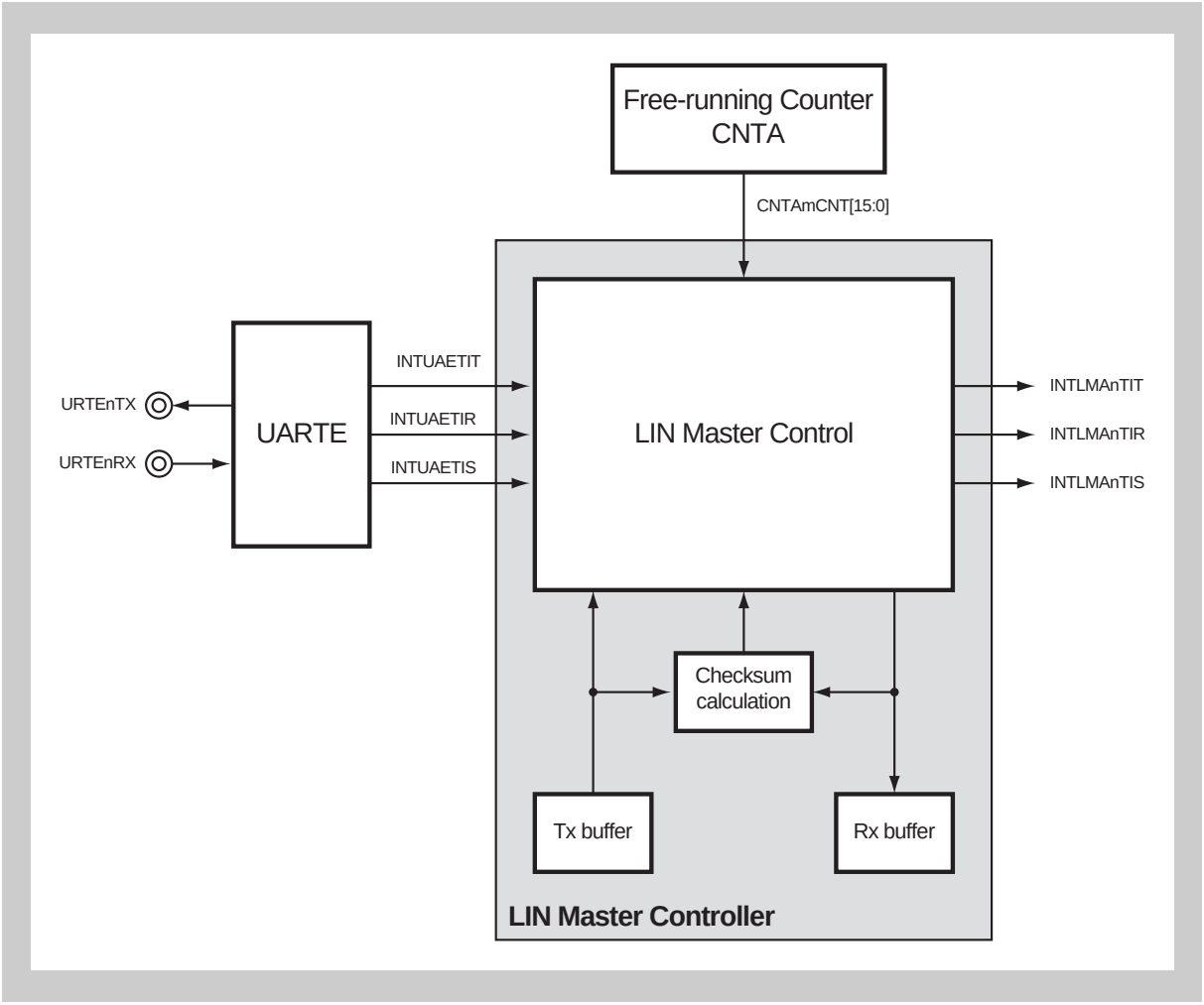


Figure 21-2 LIN master controller environment

The LIN master controller is tightly coupled to the UARTE and utilizes the UARTE as a asynchronous serial interface function with LIN capabilities.

In LIN master mode the UARTE is completely controlled by the LIN master controller, all data transfers between the Tx and Rx buffers are managed by the LIN master controller.

The UARTE interrupts signals INTUAETIT, INTUAETIR and INTUAETIS are handled by the LIN master controller, which generates the interrupt signals INTLMATIT, INTLMATIR and INTLMATIS towards the microcontroller's Interrupt Controller.

For details about UARTE, see *Chapter 20* "*Asynchronous Serial Interface E (URTE)*".

For using the scheduler and the automatic frame start function, a free-running counter is connected to the LIN master controller. See *21.2 "LIN Master Scheduler Counters (CNTA)"*.

21.4 Functional Description

The LIN Master can be set up in three different basic modes by setting the LMAnCTLL.LMAnMD[1:0] bits:

- LMAnCTLL.LMAnMD[1:0] = 00_B: UARTEn through mode
  In this mode the LMAn is bypassed and the connected UARTEn is operated as without LMAn connected.
- LMAnCTLL.LMAnMD[1:0] = 01_B: UARTEn buffer mode
  In this mode the LMAn operates as a UARTEn Rx and Tx buffer, each providing a size of 12 byte.
- LMAnCTLL.LMAnMD[1:0] = $1x_B$: LIN master mode
  In this mode the LMAn operates in combination with the UARTEn as a LIN master controller, providing Rx and Tx buffers of 12 bytes each in order to handle entire LIN frame transactions with any interaction by the CPU.

21.4.1 UART through mode

In UART through mode (LMAnCTLL.LMAnMD[1:0] = 00B) the LMAn is bypassed and the UART is used without any LMAn functions.

As this is the default LMAn mode, the UARTEn can be controlled and operated without any LMAn intervention.

Note In order to keep power consumption at a minimum, it is recommended to keep LMAnCTLH.LMAnPW = 0.

Interrupts All LMAn interrupts request are identical to the UART interrupt requests:

- transmission interrupt request: INTLMAnTIT = INTUAEnTIT
- reception interrupt request: INTLMAnTIR = INTUAEnTIR
- status interrupt request: INTLMAnTIS = INTUAEnTIS

Data transmission Data to be transmitted is written to the UARTEn transmit data register URTEnTX.

Data reception Received data is read from the UARTEn receive data register URTEnRX.

The UARTE status registers URTEnSTR0 and URTEnSTR1 provide information about data transaction status and error detections.

21.4.2 UART buffer mode

In this mode the UARTEn - LMAn combination acts a UARTEn with Rx and Tx buffers of 12 byte size each. This mode is a full-duplex mode, thus receive and transmit transactions are separately controllable and are handled simultaneously.

(1) Initialization

UARTE settings

The UARTEn must be set up as follows:

- URTEnCTL2
  - URTEnPRS[2:0], URTEnBRS[11:0]: baudrate setting
- URTEnCTL1
  - URTEnSLBM = 0: no BF reception during data reception
- URTEnCTL0
  - URTEnPW = 1: UARTEn enabled
  - URTEnCTL0.URTEnTXE = x: transmission enabled/disabled
  - URTEnCTL0.URTEnRXE = x: reception enabled/disabled
  - URTEnCTL0.URTEnSLDC = 0: no data consistency check
  - URTEnSLIT = 0: transmission interrupt request at start of transmission

All other UARTEn settings can be set as required.

LMAn settings

The LMAn must be set up as follows:

- LMAnCTLH
  - LMAnPW = 1: LMAn enabled
- LMAnCTLL
  - LMAnMD[1:0] = 01_B: UART buffer mode
  - LMAnACSE = 0: automatic checksum disabled
  - LMAnSCHE = 0: scheduler disabled
  - LMAnAFE = 0: automatic frame start function disabled
  - LMAnITMK = 0: INTLMAnTIT not masked
  - LMAnIRMK = 0: INTLMAnTIR not masked

(2) Interrupts

INTLMAnTIT

The transmission interrupt request is generated if the number of data, set up in the Tx buffer and specified by LMAnTCTLL.LMAnTLG[3:0], have been transmitted.

INTLMAnTIR

The reception interrupt request is generated if the number of data, specified by LMAnRCTLL.LMAnRLG[3:0], have been stored in the Rx buffer.
In case of continuous reception (LMAnRCTLL.LMAnRLG[3:0] = 0), INTLMAnTIR is generated after storage of the 12th data, i.e. when the Rx buffer is full.

INTLMAnTIS The status interrupt request is generated under following conditions:

- UART has detected an error during data reception:
  - parity error: URTEnSTR1.URTEnPE = 1
  - framing error: URTEnSTR1.URTEnFE = 1
  - overrun error: URTEnSTR1.URTEnOVE = 1

(3) Data transmission

For data transmission the data to be transmitted have to be written to the Tx buffer (via the LMAnTX01 to LMAnTXAB registers) and the number of bytes to be transmitted has to be specified in LMAnTCTLL.LMAnTLG[3:0] prior starting the transmission by setting the transmit request LMAnTCTLL.LMAnTRQ = 1.

The transmission interrupt request INTLMAnTIT indicates the transmission of the last data byte.

Following values are allowed for the transmit length:

- LMAnTCTLL.LMAnTLG[3:0] = 0: 12 data bytes are transmitted
- LMAnTCTLL.LMAnTLG[3:0] = 1 to 12: 1 to 12 data bytes are transmitted

Setting LMAnTCTLL.LMAnTLG[3:0] > 12 is prohibited.

The following diagram shows the principle transmission process.

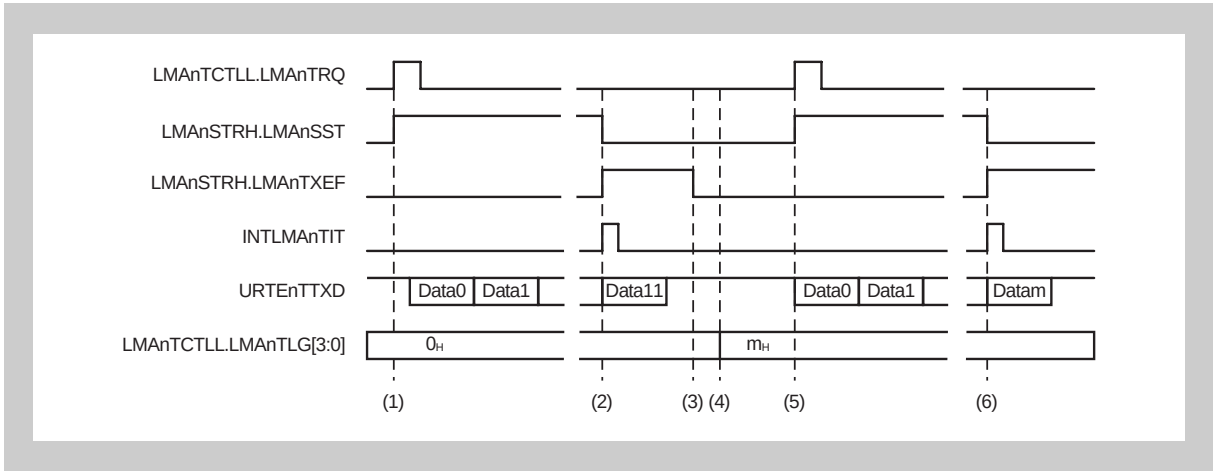


Figure 21-3 Data transmission in UART buffer mode

Precondition LMAn has been

- enabled (LMAnCTLH.LMAnPW = 1)
- set into UART buffer mode (LMAnCTLL.LMAnMD[1:0] = 01_B)
- Tx buffer empty flag LMAnSTRH.TXEF is cleared

Procedure

1. Write 12 bytes of data to the Tx buffer LMAnTX01 to LMAnTXAB and set the transmit request LMAnTCTLL.LMAnTRQ = 1. Data Tx starts afterwards.
   The start of transmission is indicated by LMAnSTRH.LMAnSST = 1.
   Since LMAnTCTLL.LMAnTLG[3:0] = 0, 12 byte will be transmitted.
2. Upon Tx start of the last data byte Data11 the transmission interrupt request INTLMAnTIT is asserted and LMAnSTRH.LMAnTXEF = 1 to indicate Tx buffer empty. LMAnSTRH.LMAnSST is cleared to 0.

3. Clear the Tx buffer empty flag by LMAnSTCH.LMAnCLTXEF = 1.
4. The next Tx buffer is prepared to transmit m byte of data by writing m data byte to the Tx buffer and set LMAnTCTLL.LMAnTLG[3:0] = $0m_H$.
5. Afterwards the next transmission is started by LMAnTCTLL.LMAnTRQ = 1.
6. Upon Tx start of the m^{th} data byte Datam the transmission interrupt request INTLMAnTIT is asserted and LMAnSTRH.LMAnTXEF = 1 to indicate Tx buffer empty. LMAnSTRH.LMAnSST is cleared to 0.

<R>

Note No errors are detected and indicated in case a transmit request is issued, though Tx buffer empty is indicated (LMAnSTRH.LMAnTXEF = 1) or the Tx length is set to incorrect values (LMAnTCTLL.LMAnTLG[3:0] > $0C_H$).

Tx abort For stopping an ongoing data transmission, a Tx abort request must be issued by LMAnTCTLH.LMAnTAB = 1. No new data from the Tx buffer is sent to the UARTE and LMAnSTRH.LMAnSST is cleared.
The UARTE completes any ongoing data transmissions. The UARTE Tx completion can be confirmed by URTEnSTR0.URTEnSST = 0.

Caution After setting an Tx abort request LMAnTCTLH.LMAnTAB = 1, a transmit interrupt request may occur. Thus mask INTLMAnTIT in the Interrupt Controller before the Tx abort request.

(4) Data reception

For data reception the number of bytes to be received has to be specified in LMAnRCTLL.LMAnRLG[3:0] prior starting the reception by setting the receive request LMAnRCTLL.LMAnRRQ = 1.
The reception interrupt request INTLMAnTIR indicates the reception of the last data byte.

Following values are allowed for the transmit length:

- LMAnRCTLL.LMAnRLG[3:0] = 0: continuous Rx mode
  In this mode received data are continuously stored to the Rx buffer without the need to set further receive requests (LMAnRCTLL.LMAnRRQ = 1).Each time 12 data bytes have been stored in the Rx buffer, a reception interrupt request INTLMAnTIR is asserted.
- LMAnRCTLL.LMAnRLG[3:0] = 1 to 12: 1 to 12 data bytes are stored in the Rx buffer

Setting LMAnRCTLL.LMAnRLG[3:0] > 12 is prohibited.

The following diagram shows the principle reception process.

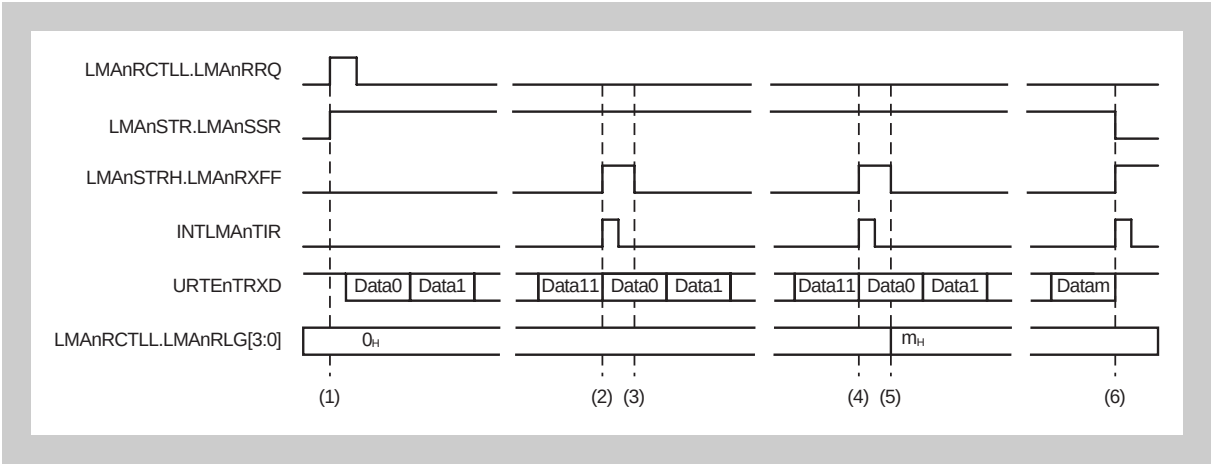


Figure 21-4 Data reception in UART buffer mode

Precondition LMAn has been

- enabled (LMAnCTLH.LMAnPW = 1)
- set into UART buffer mode (LMAnCTLL.LMAnMD[1:0] = 01_B)
- Rx buffer full flag LMAnSTRH.RXFF is cleared

Procedure

1. Set the receive request LMAnRCTLL.LMAnRRQ = 1. Data Rx starts afterwards.
   The start of reception is indicated by LMAnSTRH.LMAnSSR = 1.
   Since LMAnRCTLL.LMAnRLG[3:0] = 0, 12 byte will be received in continuous reception mode.
2. Upon Rx of the last data byte 11 the reception interrupt request INTLMAnTIR is asserted and LMAnSTRH.LMAnRXFF = 1 to indicate Rx buffer full.
   Because of continuous reception mode, Data1 is received without a new receive request.
3. The Rx buffer needs to be read via the LMAnRX00 to LMAn RXAB registers and the Rx buffer full flag is cleared by LMAnSTCH.LMAnCLRXFF = 1.
4. After reception of the next 12 data byte, only m data bytes shall be received and reception shall be stopped. Therefore Rx buffer length is set to LMAnRCTLL.LMAnRLG[3:0] = m and the Rx buffer full flag is cleared by LMAnSTCH.LMAnCLRXFF = 1.
5. If the m data bytes have been stored in the Rx buffer, reception is stopped (LMAnSTRH.LMAnSSR = 0).

Rx abort For stopping an ongoing data reception, a Rx abort request must be issued by LMAnRCTLH.LMAnRABD= 1. No new data is stored to the TR buffer, and LMAnSTRH.LMAnSSt is cleared.
The UARTE completes any ongoing data reception, but the finally received data is not stored in the Rx buffer. The UARTE Rx completion can be confirmed by URTEnSTR0.URTEnSSR = 0.

Caution After setting an Rx abort request LMAnRCTLH.LMAnRAB = 1, a receive interrupt request may occur. Thus mask INTLMAnTIR in the Interrupt Controller before the Rx abort request.

(5) UARTE Rx errors

Error detections during data reception can be initiated by the UARTE as well as by the LMAn.

UARTEn errors

If the UARTE detects a parity, framing or overrun error during data reception, the received data is stored to the Rx buffer and the assigned Rx data error flag LMAnSTRL.LMAnRXBE[11:00] is set. The number of RXBE[11:00] is associated to the data byte in the Rx buffer:

- LMAnSTRL.LMAnRXBE[00] = 1 for LMAnRX01.LMAnRX00B[7:0] error
- ...
- LMAnSTRL.LMAnRXBE[11] = 1 for LMAnRXAB.LMAnRX11B[7:0] error

If the specified number of data bytes (LMAnRCTLL.LMAnRLG[3:0]) have been stored in the Rx buffer, the status interrupt request INTLMAnTIS is asserted and reception stops (LMAnSTRH.LMAnSSR = 0), even if operating in continues reception mode (LMAnRCTLL.LMAnRLG[3:0] = 0).

Note that in this case no receive interrupt request INTLMAnTIR is generated.

The Rx error flags LMAnSTRL.LMAnRXBE[11:00] are cumulative, i.e. each reception error, that is detected until the specified number of bytes are received, set its error flag. Upon the end of reception, indicated by INTLMAnTIS, LMAnSTRL.LMAnRXBE[11:00] can be read in order to identify all Rx buffer data, that have causes error flag settings.

The following diagram shows an example in continuous reception mode (LMAnRCTLL.LMAnRLG[3:0] = 0).

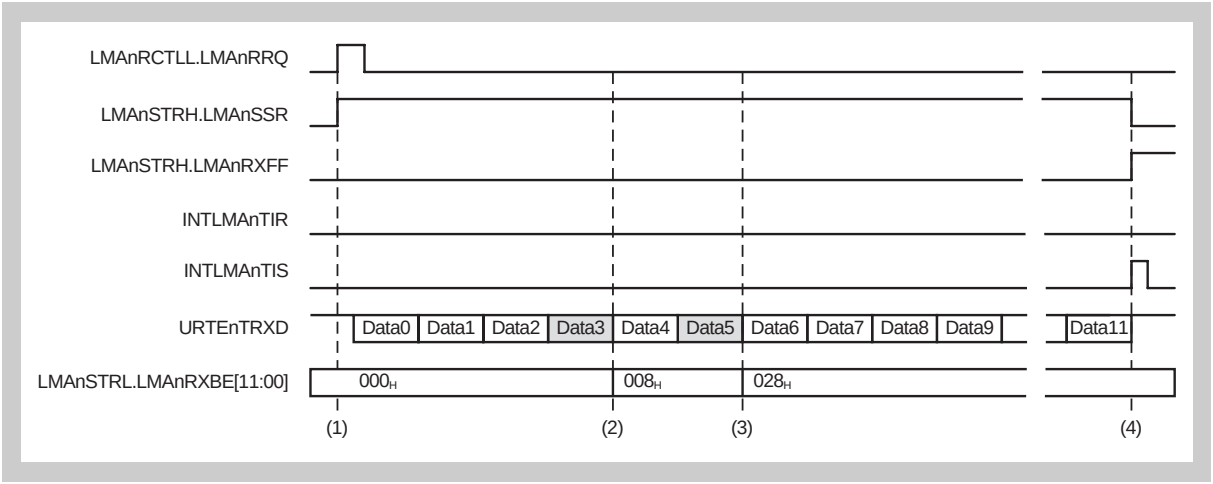


Figure 21-5 Data reception in UART buffer mode with UART reception errors

Precondition LMAn has been

- enabled (LMAnCTLH.LMAnPW = 1)
- set into UART buffer mode (LMAnCTLL.LMAnMD[1:0] = 01_B)
- Rx buffer full flag LMAnSTRH.RXFF is cleared
- Rx error flags LMAnSTRL.LMAnRXBE[11:00] has been cleared

Procedure

1. Rx is started by LMAnRCTLL.LMAnRRQ = 1.
2. During reception of Data3 the UARTE has detected an error. Data3 is stored to the Rx buffer LMAnRX23.LMAnRX03B[7:0] and the associated error flag LMAnSTRL.LMAnRXBE[03] is set, thus LMAnSTRL.LMAnRXBE[11:00] = 008_H.
3. During reception of Data5 the UARTE has detected another error. Data5 is stored to the Rx buffer LMAnRX45.LMAnRX05B[7:0] and the associated error flag LMAnSTRL.LMAnRXBE[05] is set, thus LMAnSTRL.LMAnRXBE[11:00] = 028_H.
4. After reception of the last data Data11, the status interrupt INTLMAnTIS is generated (instead of INTLMAnTIR) and the Rx process is stopped (LMAnSTRH.LMAnSSR = 0).

LMAn Rx buffer overflow error If the Rx buffer full flag LMAnSTRH.LMAnRXFF is set while new data has been received,

- the Rx buffer overflow flag LMAnSTRH.LMAnROVE is set
- the status interrupt request INTLMAnTIS is asserted
- the received data is not stored in the Rx buffer
- Rx process is stopped (LMAnSTRH.LMAnSSR = 0).

Since the received data, that is not stored in the Rx buffer, remains in the URATE's receive register URTEnRX, it is possible to read from there. However if a next data is received, this data is lost since the UARTE discards this data, i.e. it does not overwrite the URTEnRX register, but sets its overrun error flag URTEnSTR1.URTEnOVE = 1.

21.4.3 LIN master modes

In this mode the LMAn in combination with the UARTE provides a LIN master interface with automatic LIN master Tx and Rx frame transactions, including sending of Break and Sync Fields (BF and SF) to initiate a LIN master frame transaction, automatic checksum functions, scheduler and automatic frame start facilities. The separate 12-byte Tx and Rx buffers allow complete LIN frame transactions without any intervention from the CPU side.

<R> **Principle of operation**

A LIN master frame transactions is initiated by sending the frame header (BF and SF). This is done automatically when the transaction is started.

In either LIN master Tx or Rx mode the PID is be written to the Tx buffer.

In *Tx mode* the data to be transmitted (maximum 8 data bytes) has to written to the Tx buffer and the length of the Tx frame has to be specified in LMAnTCTLL.LMAnTLG[3:0].
If *automatic checksum* is enabled, the checksum is automatically calculated and appended to the Tx data bytes in the Tx buffer. In case automatic checksum is disabled, the CPU needs to calculate the checksum and to write it to the Tx buffer.
After start of the frame transmission by setting the transmit request LMAnTCTLL.LMAnTRQ = 1 transmission starts sending “BF - SF - PID - TxData bytes - Tx Checksum”.
Simultaneously the sent data is received and stored in the Rx buffer for performing the *data consistency check*, which is executed during transmission of the entire LIN frame.
If automatic checksum calculation is enabled, the checksum of the received data is automatically calculated and checked against the received checksum in the Rx buffer. In case of mismatch a checksum error is reported. If automatic checksum calculation is disabled, the checksum over the received data needs to be calculated and checked against the received checksum by the CPU.

In *Rx mode* the frame length to be received (maximum 8 data bytes) has to be specified in LMAnTCTLL.LMAnTLG[3:0].
Frame reception is also started by setting the transmit request LMAnTCTLL.LMAnTRQ = 1. The LIN master starts sending “BF - SF - PID” and then waits to receive the number of specified data bytes and finally the checksum from the LIN slave.

Note In either LIN master Tx or Rx mode the frame transaction control is done via the Tx control register LMAnTCT.

If the *scheduler* is enabled, status interrupts are generated in defined periods to ensure a minimum LIN interframe space (which is the time between two LIN frames). This minimum interframe space may be required by some slaves.
For that purpose the length of the LIN frame slot length FRSL, which spans the LIN frame plus the interframe space, is to be specified in the Tx buffer prior starting the LIN frame transaction.

If the scheduler and the *automatic frame start function* is enabled, a new LIN frame transaction is automatically started after the LIN frame slot length FRSL.

LIN frame header errors

Two modes are provided to react on errors during the LIN frame header (BF: Break Field, SF: Sync Field) transmission:

- LMAnCTLL.LMAnMD[1:0] = 10_B: LIN master mode without break in header
  Data transfers are carried on after error detections during header transmission.
- LMAnCTLL.LMAnMD[1:0] = 11_B: LIN master mode with break in header
  Data transfers are stopped after error detections during header transmission.

Checksum

Automatic checksum calculation can be performed in two ways:

- LMAnTCTLL.LMAnSLEC = 0: only the data bytes are used to calculate the checksum (classic checksum)
- LMAnTCTLL.LMAnSLEC = 1: the PID and the data bytes are used to calculate the checksum (enhanced checksum)

LIN frame length

The number of bytes to be transferred, i.e. the length of the LIN frame, needs to be specified in LMAnTCTLL.LMAnTLG[3:0].
The LMAnTCTLL.LMAnTLG[3:0] value includes the number of data bytes (maximum 8), the PID and the checksum. Thus

LMAnTCTLL.LMAnTLG[3:0] = 2 to 10

All other values are prohibited.

Tx/Rx abort

For stopping an ongoing data transaction, a Tx abort request must be issued by LMAnTCTLH.LMAnTAB = 1.
No new data from the Tx buffer is sent to the UARTE respectively received from the UARTE and stored in the Rx buffer.LMAnSTRH.LMAnSST is cleared. The UARTE completes any ongoing data transmissions. The UARTE data transmission/reception completion can be confirmed by URTEnSTR0.URTEnSST =URTEnSTR0.URTEnSST = 0.

Caution After setting an Tx abort request LMAnTCTLH.LMAnTAB = 1, a transmit, reception or status interrupt request may occur. Thus mask INTLMAnTIT, INTLMAnTIR, and INTLMAnTIS in the Interrupt Controller before the Tx abort request.

(1) Initialization

CNTAn settings If the scheduler is to be used, the scheduler counter must be set up as follows:

- CNTAmCTL
  - CNTAmPW = 1: counter enabled
- CNTAmCFG
  - CNTAmPRS[15:00]: division ratio

UARTE settings The UARTEn must be set up as follows:

- URTEnCTL2
  - URTEnPRS[2:0], URTEnBRS[11:0]: baudrate setting
- URTEnCTL1
  - URTEnSLBM = 0: no BF reception during data reception
  - URTEnBLG[2:0] = 0: BF bit length
  - URTEnCLG = 1: 8-bit data
  - URTEnSLP[1:0] = 00_B: no parity
  - URTEnTDL = 0: no Tx data inversion
  - URTEnRDL = 0: no Rx data inversion
  - URTEnSLG = 0: 1 stop bit
  - URTEnSLD = 1: LSB first
  - URTEnSLIT = 0: transmission interrupt request at start of transmission
- URTEnCTL0
  - URTEnPW = 1: UARTEn enabled
  - URTEnCTL0.URTEnTXE = 1: transmission enabled
  - URTEnCTL0.URTEnRXE = 1: reception enabled
  - URTEnCTL0.URTEnSLDC = 1: data consistency check enabled

LMAn settings The LMAn must be set up as follows:

- LMAnCTLH
  - LMAnPW = 1: LMAn enabled
- LMAnCTLL
  - LMAnMD[1:0] = $1x_B$: LIN master mode
  - LMAnACSE = x: automatic checksum enabled/disabled
  - LMAnSCHE = x: scheduler enabled/disabled
  - LMAnAFE = x: automatic frame start function enabled/disabled
  - LMAnITMK = x: INTLMAnTIT masked/not masked
  - LMAnIRMK = x: INTLMAnRIT masked/not masked

(2) Interrupts

Since both transmission and reception are involved during a LIN master frame transaction, transmit (INTLMAnTIT) and receive (INTLMAnTIR) interrupts are generated in Tx as well as in Rx mode.

Note Both interrupt requests can be separately suppressed:

- LMAnCTLL.ITMK = 1: INTLMAnTIT is masked and will not be generated
- LMAnCTLL.IRMK = 1: INTLMAnTIR is masked and will not be generated

INTLMAnTIT

- in Tx mode (LMAnTCTLL.LMAnSLRT = 0)
  INTLMAnTIT is generated if the number of data, specified by LMAnTCTLL.LMAnTLG[3:0], have been transmitted.
- in Rx mode (LMAnTCTLL.LMAnSLRT = 1)
  if SF has been transmitted

INTLMAnTIR

- in Tx mode (LMAnTCTLL.LMAnSLRT = 0)
  Generation of INTLMAnTIR depends also on the auto-checksum function. If the checksum has been received for checksum control and
  - auto-checksum function is disabled (LMAnCTLL.LMAnACSE = 0), INTLMAnTIR is always asserted.
  - auto-checksum function is enabled (LMAnCTLL.LMAnACSE = 1), INTLMAnTIR is asserted, if the received checksum matches the automatically calculated checksum. In case of a mismatch a checksum error is indicated (LMAnSTRH.LMAnFCSE = 1) and the status interrupt request INTLMAnTIS is generated instead.
- in Rx mode (LMAnTCTLL.LMAnSLRT = 1)
  INTLMAnTIR is generated if the number of data, specified by LMAnRCTLL.LMAnRLG[3:0], have been stored in the Rx buffer.

INTLMAnTIS The status interrupt request is generated under various conditions:

- UARTE has detected a framing error.
- UARTE has detected an overrun error.
- UARTE has detected a data consistency error.
- UARTE has detected a BF transmission error.
- UARTE has detected a SF transmission error.
- LMAn has detected an auto-checksum error.
- LMAn has detected a buffer preparation error.
- LMAn scheduler ready event occurred.

(3) Data transmission

The LIN master Tx mode is selected by LMAnCTLL.LMAnMD[1:0] = $1x_B$ and LMAnTCTLL.LMAnSLRT = 0.

For transmission of a LIN master frame the Tx buffer has to be prepared in the following format prior starting the frame transmission:

Table 21-13 Tx buffer preparation in LIN master Tx mode

Tx buffer register		Tx buffer for 8 data byte	Tx buffer for 5 data byte
LMAnTXAB.	LMAnTX11B[7:0]	FRSLH[a]	FRSLH[a]
	LMAnTX10B[7:0]	FRSLL[a]	FRSLL[a]
LMAnTX89.	LMAnTX9B[7:0]	Tx checksum[b]	–
	LMAnTX8B[7:0]	TxData7	–
LMAnTX67.	LMAnTX7B[7:0]	TxData6	–
	LMAnTX6B[7:0]	TxData5	Tx checksum[b]
LMAnTX45.	LMAnTX5B[7:0]	TxData4	TxData4
	LMAnTX4B[7:0]	TxData3	TxData3
LMAnTX23.	LMAnTX3B[7:0]	TxData2	TxData2
	LMAnTX2B[7:0]	TxData1	TxData1
LMAnTX01.	LMAnTX1B[7:0]	TxData0	TxData0
	LMAnTX0B[7:0]	PID	PID

a) The frame slot length FRSLL/FRSLH is only effective if the scheduler is enabled (LMAnCTLL.LMAnSCHE = 1).

b) The Tx checksum is automatically stored, if automatic checksum is enabled (LMAnCTLL.LMAnACSE = 1). Otherwise the checksum needs to be calculated stored by software.

After starting the LIN frame transmission the sent data is stored in the Rx buffer for checksum confirmation. When the entire frame has been transmitted, the Rx buffer looks as follows:

Table 21-14 Rx buffer after LIN frame transmission

Rx buffer register		Rx buffer for 8 data byte	Rx buffer for 5 data byte
LMAnRxAB.	LMAnRx11B[7:0]	–	–
	LMAnRx10B[7:0]	–	–
LMAnRx89.	LMAnRx9B[7:0]	Rx checksum	–
	LMAnRx8B[7:0]	RxData7	–
LMAnRx67.	LMAnRx7B[7:0]	RxData6	–
	LMAnRx6B[7:0]	RxData5	Rx checksum
LMAnRx45.	LMAnRx5B[7:0]	RxData4	RxData4
	LMAnRx4B[7:0]	RxData3	RxData3
LMAnRx23.	LMAnRx3B[7:0]	RxData2	RxData2
	LMAnRx2B[7:0]	RxData1	RxData1
LMAnRx01.	LMAnRx1B[7:0]	RxData0	RxData0
	LMAnRx0B[7:0]	PID	PID

If automatic checksum is enabled (LMAnCTLL.LMAnACSE = 1), the checksum (as selected by LMAnTCTLL.LMAnSLEC) is calculated and compared with the received checksum. Otherwise the checksum needs to be calculated and compared by software.

The following diagram shows the principle LIN frame transmission process with the maximum of 8 data byte.

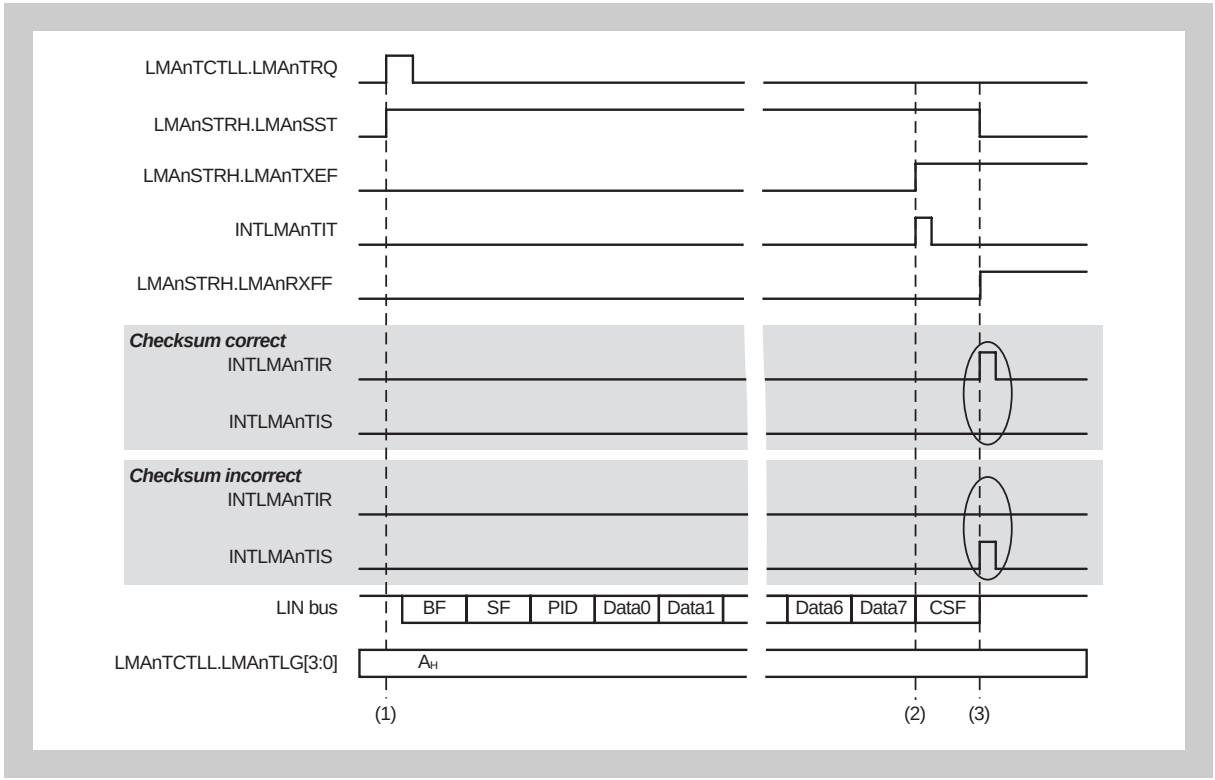


Figure 21-6 LIN frame transmission in LIN master mode

Precondition LMAn has been

- enabled (LMAnCTLH.LMAnPW = 1)
- set into LIN master Tx mode (LMAnCTLL.LMAnMD[1:0] = $1x_B$, LMAnTCTLL.LMAnSLRT = 0)
- neither scheduler and, thus, nor automatic frame start function used (LMAnSCHE and LMAnAFE of LMAnCTLL register both cleared)
- Tx buffer empty flag LMAnSTRH.TXEF is cleared
- Rx buffer full flag LMAnSTRH.RXFF is cleared

Procedure

1. Write the PID to Tx buffer LMAnTX01.LMAnTX00B[7:0] and 8 bytes of data to the Tx buffer LMAnTX01.LMAn01B[7:0] to LMAnTX89.LMAn08B[7:0].
<R> If automatic checksum is disabled (LMAnCTLL.LMAnACSE = 0), the CPU must calculate the CSF (checksum field) and add it to the Tx buffer. If automatic checksum is enabled, the CSF will be calculated and added automatically.
Set the frame length LMAnTCTLL.LMAnTLG[3:0] = A_H (10 = PID + 8 data bytes + CSF).
Start frame transmission by LMAnTCTLL.LMAnTRQ = 1, and transmission start is indicated by LMAnSTRH.LMAnSST = 1.
2. Upon start of the Checksum Field (CSF) transmission, Tx buffer empty is indicated by LMAnSTRH.LMAnTXEF = 1 and the transmit interrupt request INTLMAnTIT is asserted.
3. After the checksum field CSF has been transmitted, the Rx buffer full flag LMAnSTRH.LMAnRXFF is set and checksum control is performed, provided automatic checksum calculation is enabled

(LMAnCTLL.LMAnACSE = 1.
If the checksum is correct, the receive interrupt request INTLMAnTIR is asserted.
If the checksum is incorrect, the status INTLMAnTIS instead of the receive interrupt request is asserted.

(4) Data reception

The LIN master Rx mode is selected by LMAnCTLL.LMAnMD[1:0] = $1x_B$ and LMAnTCTLL.LMAnSLRT = 1.

For data reception in LIN master mode the Tx buffer has to be prepared in the following format prior starting the frame reception:

Table 21-15 Tx buffer preparation in LIN master Rx mode

Tx buffer register		Tx buffer
LMAnTXAB.	LMAnTX11B[7:0]	FRSLH[a]
	LMAnTX10B[7:0]	FRSLL[a]
LMAnTX89.	LMAnTX9B[7:0]	–
	LMAnTX8B[7:0]	–
LMAnTX67.	LMAnTX7B[7:0]	–
	LMAnTX6B[7:0]	–
LMAnTX45.	LMAnTX5B[7:0]	–
	LMAnTX4B[7:0]	–
LMAnTX23.	LMAnTX3B[7:0]	–
	LMAnTX2B[7:0]	–
LMAnTX01.	LMAnTX1B[7:0]	–
	LMAnTX0B[7:0]	PID

a) The frame slot length FRSLL/FRSLH is only effective if the scheduler is enabled (LMAnCTLL.LMAnSCHE = 1).

After starting the LIN frame reception the LIN master frame header (BF, SF, PID) is transmitted to the slaves and the data, received from the slave afterwards, is stored in the Rx buffer. When the entire frame has been transmitted, the Rx buffer looks as follows:

Table 21-16 LIN master Rx buffer after LIN frame reception (1/2)

Rx buffer register		Rx buffer for 8 data byte	Rx buffer for 5 data byte
LMAnRxAB.	LMAnRx11B[7:0]	–	–
	LMAnRx10B[7:0]	–	–
LMAnRx89.	LMAnRx9B[7:0]	Rx checksum	–
	LMAnRx8B[7:0]	RxData7	–
LMAnRx67.	LMAnRx7B[7:0]	RxData6	–
	LMAnRx6B[7:0]	RxData5	Rx checksum
LMAnRx45.	LMAnRx5B[7:0]	RxData4	RxData4
	LMAnRx4B[7:0]	RxData3	RxData3

Table 21-16 LIN master Rx buffer after LIN frame reception (2/2)

Rx buffer register		Rx buffer for 8 data byte	Rx buffer for 5 data byte
LMAnRx23.	LMAnRx3B[7:0]	RxData2	RxData2
	LMAnRx2B[7:0]	RxData1	RxData1
LMAnRx01.	LMAnRx1B[7:0]	RxData0	RxData0
	LMAnRx0B[7:0]	PID	PID

If automatic checksum is enabled (LMAnCTLL.LMAnACSE = 1), the checksum (as selected by LMAnTCTLL.LMAnSLEC) is calculated and compared with the received checksum. Otherwise the checksum needs to be calculated and compared by software.

The following diagram shows the principle LIN frame reception process with the maximum of 8 data byte.

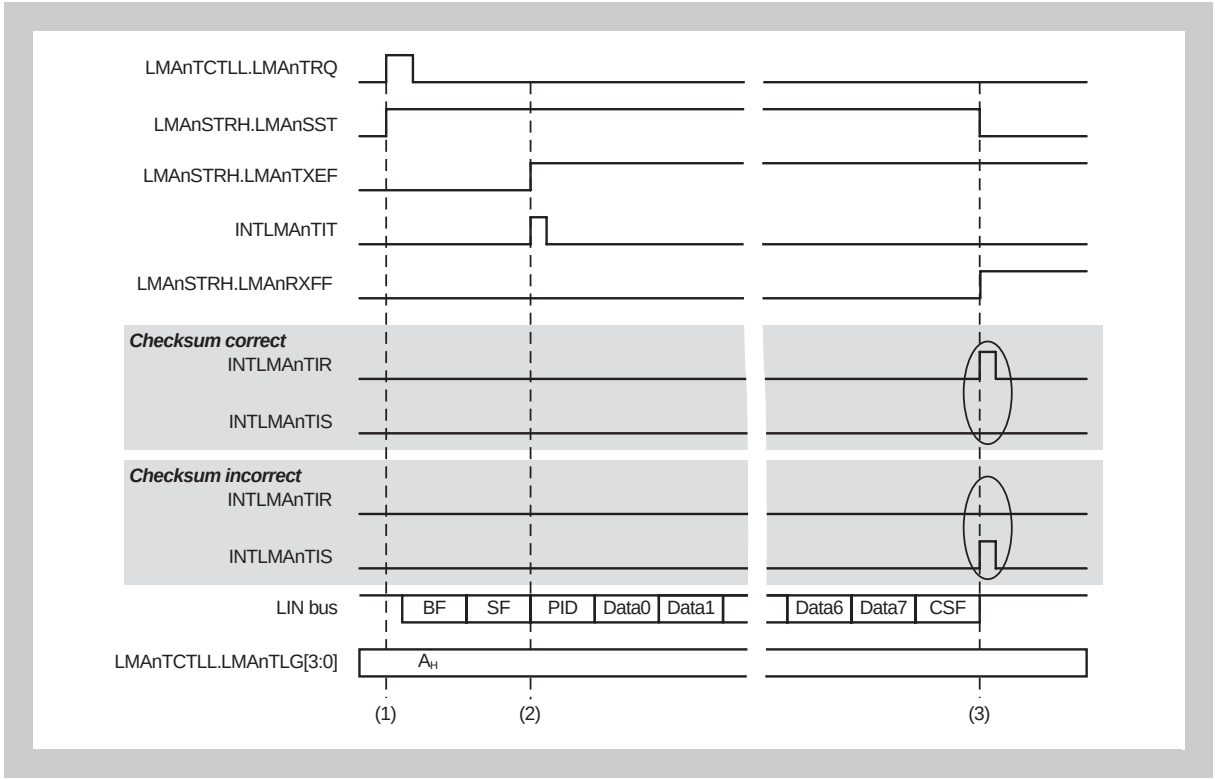


Figure 21-7 LIN frame reception in LIN master mode

Precondition LMAn has been

- enabled (LMAnCTLH.LMAnPW = 1)
- set into LIN master Rx mode (LMAnCTLL.LMAnMD[1:0] = $1x_B$, LMAnTCTLL.LMAnSLRT = 1)
- neither scheduler and, thus, nor automatic frame start function used (LMAnSCHE and LMAnAFE of LMAnCTLL register both cleared)
- Tx buffer empty flag LMAnSTRH.TXEF is cleared
- Rx buffer full flag LMAnSTRH.RXFF is cleared

Procedure

1. Write the PID to Tx buffer LMAnTX01.LMAnTX00B[7:0].
   The number of data
   Start frame reception by LMAnTCTLL.LMAnTRQ = 1, and transmission start of the frame header is indicated by LMAnSTRH.LMAnSST = 1.
2. Upon start of the PID transmission, Tx buffer empty is indicated by LMAnSTRH.LMAnTXEF = 1 and the transmit interrupt request INTLMAnTIT is asserted.
3. After the checksum field CSF has been transmitted, the Rx buffer full flag LMAnSTRH.LMAnRXFF is set and checksum control is performed, provided automatic checksum calculation is enabled (LMAnCTLL.LMAnACSE = 1.
   If the checksum is correct, the receive interrupt request INTLMAnTIR is asserted.
   If the checksum is incorrect, the status INTLMAnTIS instead of the receive interrupt request is asserted.

(5) LIN master mode transaction errors

During LIN master mode transactions the UARTEn as well as the LMAn can detected and indicate various transaction errors.

Table 21-17 LIN master mode transaction errors

Detection module	Error	Indicator	Error cause
UARTE	Framing error	URTEnSTR1.URTEnFE = 1	No stop bit was detected after reception of the 8th bit of the SF, PID, data, or CSF (checksum) byte. A framing error generates also a data consistency error.
	Overrun error	URTEnSTR1.URTEnOVE = 1	New data received, while the UARTE receive register URTEnRX holds data, that has not been stored in the Rx buffer.
	Data consistency error	URTEnSTR1.URTEnDCE = 1	Data sent during transmission is erroneous. A data consistency error during BF transmission is indicated as a BF transmission error. A data consistency error during SF transmission is indicated as a SF transmission error. Note that any framing error generates also a data consistency error.
	BF transmission error	URTEnSTR1.URTEnBSF = 0 LMAnSTRH.LMANBFE = 1	Data consistency occurs during BF Tx
	SF transmission error	URTEnSTR1.URTEnDCE = 1 LMAnSTRH.LMANSFE = 1	Data consistency error occurred during SF transmission.
LMAn	Auto-checksum error	LMAnSTRH.LMAnFCSE = 1	Calculated checksum does not match the received checksum after completion of a LIN frame transaction. provided the auto-checksum function is enabled by LMAnCTLL.LMAnACSE = 1.
	Rx/Tx buffer preparation error	LMAnSTRH.LMAnPIE = 1	LIN frame transaction is started (by LMAnTCTLL.LMAnTRQ = 1 or in auto frame start mode), while the Tx and Rx buffer are not set up correctly, i.e. • Tx buffer empty (LMAnSTRH.LMAnTXEF = 1) • Rx buffer full (LMAnSTRH.LMAnRXFF = 1) • transmit length incorrect (LMAnTCTLL.LMAnTLG[3:0] = 0, 1, 11 to 15)

The procedure taking place upon an UARTE error detection depends on the mode the LMAn is operating in and the type of erroneous data.

The main difference between LIN master mode with and without break in header refers to the different behaviour, when a data consistency check error is detected in the LIN frame header (BF/SF) in Rx mode: “with break in header” stops any further transactions, while “without break in header” continuous transactions.

- LIN master mode with break in header

  If a *data consistency check error* was detected in any type of data, further transactions are stopped, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated.

In case of *framing/overrun error* detections during transmission of the BF/SF transactions are continued.
If PID/data/checksum are received with framing/overrun errors in Rx mode, further transactions are stopped, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated.

- LIN master mode without break in header

  If a *data consistency check error* was detected in PID/data/checksum, or during transmission of BF/SF, further transactions are stopped, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated. If data inconsistency during BF/SF in Rx mode, transactions continue, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated.

  In case of *framing/overrun error* detections during transmission of the BF or SF transactions are continued.
  If PID/data/Checksum are received with framing/overrun errors in Rx mode, further transactions are stopped, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated.

The following table summarizes these procedures.

Table 21-18 UARTE errors in LIN master mode

Rx/Tx mode	Framing error[a]/overrun error in		Data consistency check error in	
	BF/SF[b]	PID/data/CSF	BF / SF	PID / Data / CSF
LIN master mode with break in header (LMAnCTLL.LMAnMD[1:0] = 11_B)				
Tx mode (LMAnTCTLL. LMAnSLRT = 0)	Transaction continued		• Transaction stopped at STOP bit of erroneous field • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted	
Rx mode (LMAnTCTLL. LMAnSLRT = 1)	Transaction continued	• Transaction stopped at STOP bit of erroneous field • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted		
LIN master mode without break in header (LMAnCTLL.LMAnMD[1:0] = 10_B)				
Tx mode (LMAnTCTLL. LMAnSLRT = 0)	Transaction continued		• Transaction stopped after PID transmitted • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted ater PID transmitted	• Transaction stopped at STOP bit of erroneous field • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted
Rx mode (LMAnTCTLL. LMAnSLRT = 1)	Transaction continued	• Transaction stopped at STOP bit of erroneous field • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted	• Transaction continued • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted after CSF reception completed	

a) A framing error during transmission of any type of data a data consistency error is detected simultaneously (in Tx mode: BF/SF/PID/data/CSF are transmitted, in Rx mode: BF/SF/PID/ are transmitted).

b) Any error detection in BF or SF sets the respective error flag LMAnSTRH.LMAnBFE respectively LMAnSTRH.LMAnSFE.

21.4.4 Automatic checksum function

The automatic checksum function allows to automatically generate respectively control checksums.

The automatic checksum function is control by:

- LMAnCTLL.LMAnACSE = 0: automatic checksum function disabled
- LMAnCTLL.LMAnACSE = 1: automatic checksum function enabled

In LMAnCTLL.LMAnACSE = 0: automatic checksum function is enabled, the automatic checksum function performs the following:

Tx mode In Tx mode (LMAnTCTLL.LMAnSLRT = 0), the checksum is automatically calculated and appended to the transmit data in the Tx buffer upon LIN frame transaction start.

<R>

Rx mode After reception completion the checksum is calculated from the received data and automatically compared to the received checksum, stored in the Rx buffer. In case both match, the receive interrupt request INTLMAnTIR is generated. If they don't match, the status interrupt request INTLMAnTIS is generated and the checksum error flag LMAnSTRH.LMAnFCSE is set.

Checksum format The data incorporated in the calculation of the checksum in automatic checksum mode can be chosen:

- LMAnTCTLL.LMAnSLEC = 0: classic checksum
  Only the data bytes, stored in the Tx or Rx buffer, are used to calculate the checksum.
- LMAnTCTLL.LMAnSLEC = 1: enhanced checksum
  The data bytes and the PID, stored in the Tx or Rx buffer, are used to calculate the checksum.

21.4.5 Scheduler

The scheduler allows to generate status interrupts INTLMAnTIS in defined time distances. INTLMAnTIS can be used to initiate the next LIN master frame transaction.
By this a minimum LIN interframe space (which is the time between two LIN frames) can be ensured. This minimum interframe space may be required by some slaves.

For that purpose the LIN frame slot length FRSL, which spans the LIN frame plus the interframe space, is to be specified in the Tx buffer prior starting the LIN frame transaction.

Scheduler counter

The scheduler makes use of a scheduler counter CNTAm. CNTAm is a free-running counter. Its count clock can be selected by the prescaler CNTAmCFG.CNTAmPS[15:0].
For details about the scheduler counter CNTA, see *21.2 “LIN Master Scheduler Counters (CNTA)”*.

(1) Scheduler operation

Note

Before any LIN master frame transaction is started with scheduler, the employed scheduler counter CNTAm has to be enabled (CNTAmCTL.CNTAmPWR = 1) and its prescaler has to be set up (CNTAmCFG.CNTAmPS[15:0]). CNTAmCFG.CNTAmPS[15:0] determines the scheduler count clock SCHECLK. The scheduler clocks are counting up the scheduler counters count value CNTAmCNT.

The scheduler needs to be enabled by LMAnCTLL.LMAnSCHE = 1.

Before starting a LIN master frame transaction (LMAnTCTLL.LMAnTRQ = 1) with scheduler function, the length of the 16-bit LIN frame slot length FRSL[15:0] has to be written to the Tx buffer:

- LMAnTXAB.LMAnTX10B[7:0] = FRSLL = FRSL[7:0]
- LMAnTXAB.LMAnTX11B[7:0] = FRSLH = FRSL[15:8]

FRSL[15:0] defines the number of scheduler clocks SCHECLK for the frame slot length.

If the LIN master frame transaction is started, the current value of the scheduler counter CNTAmCNT[15:0] is added to the defined frame slot length FRSL[15:0] and stored to the compare register LMAnCMPL.LMAnCMP[15:0].

The scheduler counter value CNTAmCNT[15:0] is continuously compared with the compare register LMAnCMP[15:0]. If both match, the status interrupt request INTLMAnTIS is generated and the scheduler ready flag LMAnSTRH.LMAnSRF is set.

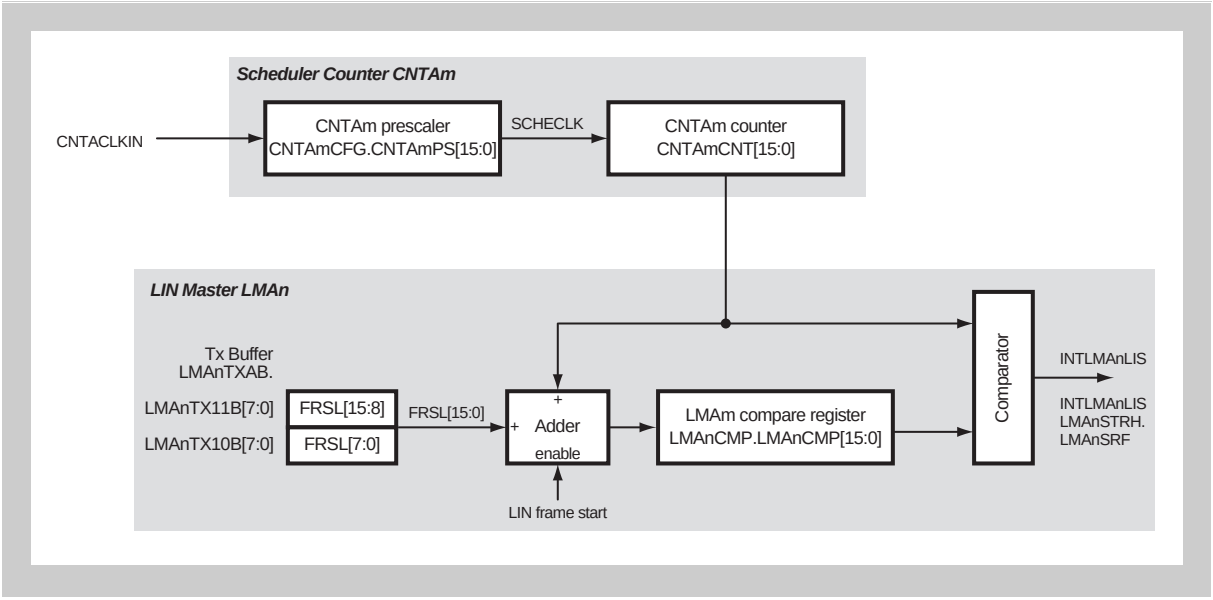


Figure 21-8 Scheduler functional overview

Interrupt handling

Upon occurrence of the status interrupt INTLMAnTIS transaction of a next LIN master frame can be started immediately, provided the frame slot length FRSL includes also a minimum interframe space. Thus the receive and transmit interrupts may not be necessary to be processed and could be suppressed by masking LMAnCTLL.LMAnITMK = LMAnCTLL.LMAnIRMK = 1.

The diagram below shows the principle timing of a LIN frame transmission with use of the scheduler.

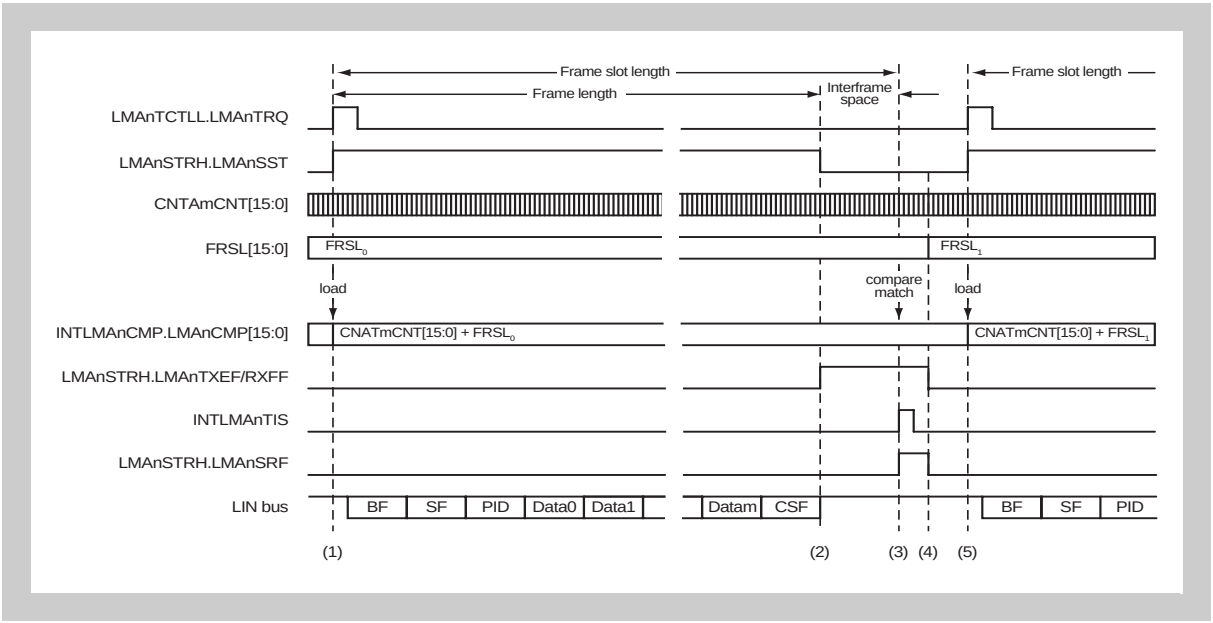


Figure 21-9 LIN frame transaction with scheduler

Precondition

The scheduler counter is operating and scheduler clock frequency has been set up.
LMAn has been

- enabled (LMAnCTLH.LMAnPW = 1)
- set into LIN master mode (LMAnCTLL.LMAnMD[1:0] = $1x_B$)
- scheduler is enabled (LMAnCTLL.LMAnSCHE = 1)
- automatic frame start function is disabled (LMAnCTLL.LMAnAFE = 0)
- Tx buffer empty flag LMAnSTRH.TXEF and Rx buffer full flag LMAnSTRH.RXFF are cleared
- Tx buffer has been set up correctly with frame slot length $FRSL_0$

Procedure

1. Start frame transmission by LMAnTCTLL.LMAnTRQ = 1, and transmission start is indicated by LMAnSTRH.LMAnSST = 1.
   Sum of the current schedule counter value CNTAmCNT[15:0] and the frame slot length $FRSL_0$ is stored to LMAnCMP.CMP[15:0].
2. After transmission of the checksum field CSF, i.e. after the frame length, the Tx buffer is indicated as empty (LMAnSTRH.TXEF = 1) and the Rx buffer as full (LMAnSTRH.RXFF = 1).
3. Upon match of the scheduler counter CNTAmCNT[15:0] and LMAnCMP.LMAnCMP[15:0] the status interrupt request INTLMAnTIS indicates the end of the frame slot length and the scheduler ready flag LMAnSTRH.LMAnSRF is set.
   If a sufficient interframe space was regarded, when defining the frame slot length $FRSL_0$, the next frame transaction could already be started at this point in time.
4. The buffer status flags and the scheduler ready flag are cleared (LMAnCLTXEF = LMAnCLRXFF = LMAnCLSRF of LMAnSTCH register set to 1).
   The Tx buffer and its control register LMAnTCTL are prepared for the next frame transaction, with the next frame slot length $FRSL_1$.
5. Next frame transaction is started by LMAnTCTLL.LMAnTRQ = 1, and transmission start is indicated by LMAnSTRH.LMAnSST = 1.
   Sum of the current schedule counter value CNTAmCNT[15:0] and the frame slot length $FRSL_1$ is stored to LMAnCMP.CMP[15:0].

(2) Scheduler operation with automatic frame start function

<R>

The automatic frame start function, in combination with the scheduler, reduces the jitter that occurs when a LIN frame starts by allowing the CPU to make all preparations within the interframe space. The next LIN frame transaction is started automatically after the interframe space, thus maintaining a maximum data transmission performance on the bus.

The automatic frame start function needs to be enabled by LMAnCTLL.LMAnAFE = 1.
Note that the scheduler has to be enabled as well by LMAnCTLL.LMAnSCHE = 1.

In order to start LIN frame transactions with automatic frame start function the first Tx request bit LMAnTCTLL.LMAnFRQ has to be set to 1 in addition to the first Tx request LMAnTCTLL.LMAnTRQ. After LMAnTRQ has returned to 0, it can be set again to 1, although the frame transfer is ongoing (LMAnSTRH.LMAnSST = 1). Thus the next frame transaction is started immediately after the next interframe space has passed.

During the interframe space all preparations (Rx/Tx buffer setup, etc.) have to be finished for the next frame transaction. These preparations can be triggered by the reception interrupt request INTLMAnTIR.

Caution The reception interrupt request INTLMAnTIR must be used to initiate frame preparations in Tx as well as in Rx mode.

The diagram below shows the principle timing of a LIN frame transmission with use of the scheduler and automatic frame start function.

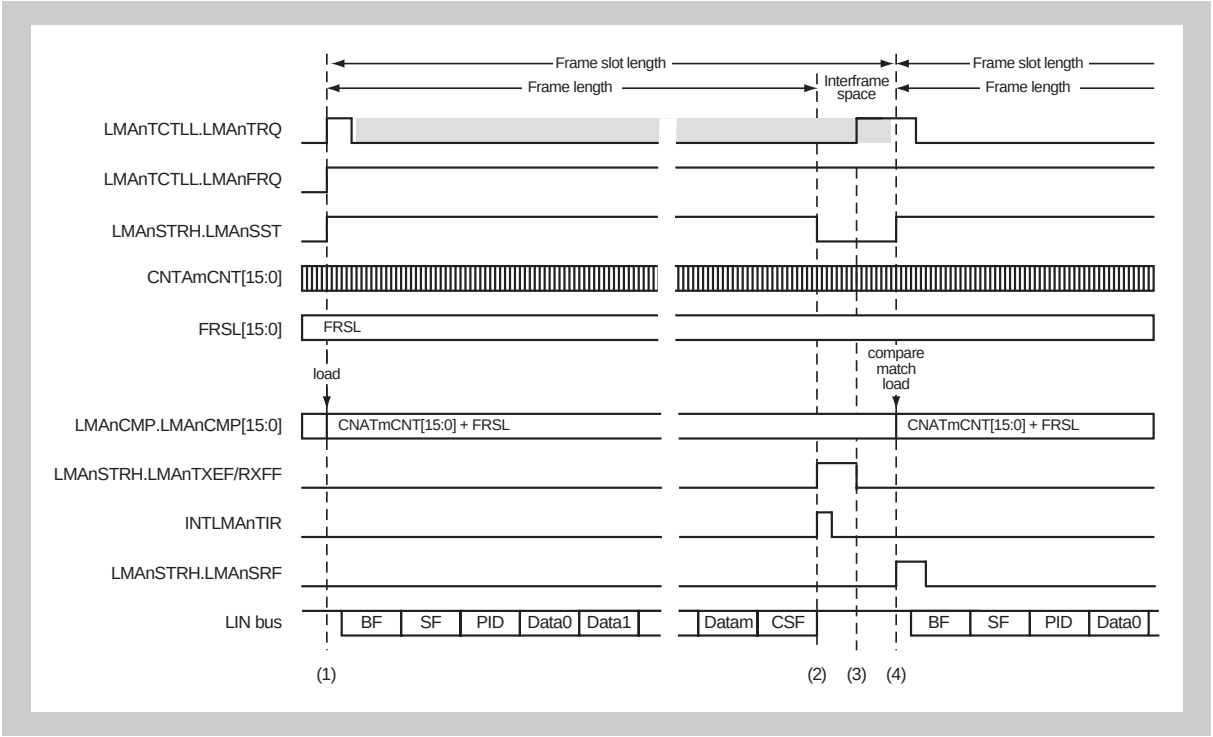


Figure 21-10 LIN frame transmission with scheduler and automatic frame start function

Precondition The scheduler counter is operating and scheduler clock frequency has been set up.
LMAn has been

- enabled (LMAnCTLH.LMAnPW = 1)
- set into LIN master mode (LMAnCTLL.LMAnMD[1:0] = $1x_B$)
- scheduler is enabled (LMAnCTLL.LMAnSCHE = 1)
- automatic frame start function is enabled (LMAnCTLL.LMAnAFE = 1)
- Tx buffer empty flag LMAnSTRH.TXEF and Rx buffer full flag LMAnSTRH.RXFF are cleared
- Tx buffer has been set up correctly with frame slot length FRSL

Procedure

1. Start first transmission with automatic frame start function by LMAnTCTLL.LMAnTRQ = 1 and LMAnTCTLL.LMAnFRQ = 1.Transmission start is indicated by LMAnSTRH.LMAnSST = 1.
   Sum of the current schedule counter value CNTAmCNT[15:0] and the frame slot length FRSL is stored to LMAnCMP.CMP[15:0].

After LMAnTCTLL.LMAnTRQ has returned to 0, the next transmission request can be set. At the latest it needs to be set before the interframe space has ended, i.e. in the gray area.
However in the diagram LMAnTRQ is set in the interframe space.

2. After transmission of the checksum field CSF, i.e. after the frame length, the Tx buffer is indicated as empty (LMAnSTRH.TXEF = 1) and the Rx buffer as full (LMAnSTRH.RXFF = 1). The reception interrupt request INTLMAnTIR is asserted.
3. All preparations are performed for the next frame transmission.
The buffer status flags and the scheduler ready flag are cleared (LMAnCLTXEF = LMAnCLRXFF = LMAnCLSRF of LMAnSTCH register set to 1).
The next Tx request is set (LMAnTCTLL.LMAnTRQ = 1).
4. Upon match of the scheduler counter CNTAmCNT[15:0] and LMAnCMP.LMAnCMP[15:0], next frame transmission is started.

Preparation error

If the next frame is not completely and correctly prepared,

- LMAnTCTLL.LMAnTRQ and LMAnSTRH.LMAnSST are cleared and frame transaction is not started
- LMAnCMP.LMAnCMP[15:0] is loaded with CNATmCNT[15:0] + FRSL
- the preparation incomplete error flag LMAnSTRH.LMAnPIE is set
- the status interrupt request LMAnTIS is asserted

If frame preparation is completed with start of the next frame, i.e. if CNATmCNT[15:0] = LMAnCMP.LMAnCMP[15:0], frame transmission starts. Otherwise the next LMAnTIS interrupt request is asserted to indicate another preparation incomplete error.

21.5 LMAn Registers

This section contains a description of all LMA registers.

21.5.1 LMAn register overview

LMAn is controlled and operated by the following registers:

Table 21-19 LMAn register overview

Register name	Symbol	Address
Control and status registers:		
Control register L	LMAnCTLL	<LMAn_base> + 80_H
Control register H	LMAnCTLH	<LMAn_base> + 84_H
Status register L	LMAnSTRL	<LMAn_base> + 88_H
Status register H	LMAnSTRH	<LMAn_base> + $8C_H$
Status clear register L	LMAnSTCL	<LMAn_base> + 90_H
Status clear register H	LMAnSTCH	<LMAn_base> + 94_H
Compare register L	LMAnCMPL	<LMAn_base> + $9C_H$
Tx control register L	LMAnTCTLL	<LMAn_base> + $D8_H$
Tx control register H	LMAnTCTLH	<LMAn_base> + DC_H
Rx control register L	LMAnRCTLL	<LMAn_base> + $F8_H$
Rx control register H	LMAnRCTLH	<LMAn_base> + FC_H
Tx buffer registers:		
Tx buffer register 01	LMAnTX01	<LMAn_base> + $C0_H$
Tx buffer register 23	LMAnTX23	<LMAn_base> + $C4_H$
Tx buffer register 45	LMAnTX45	<LMAn_base> + $C8_H$
Tx buffer register 67	LMAnTX67	<LMAn_base> + CC_H
Tx buffer register 89	LMAnTX89	<LMAn_base> + $D0_H$
Tx buffer register AB	LMAnTXAB	<LMAn_base> + $D4_H$
Rx buffer registers:		
Rx buffer register 01	LMAnRX01	<LMAn_base> + $E0_H$
Rx buffer register 23	LMAnRX23	<LMAn_base> + $E4_H$
Rx buffer register 45	LMAnRX45	<LMAn_base> + $E8_H$
Rx buffer register 67	LMAnRX67	<LMAn_base> + EC_H
Rx buffer register 89	LMAnRX89	<LMAn_base> + $F0_H$
Rx buffer register AB	LMAnRXAB	<LMAn_base> + $F4_H$

<LMAn_base> The base addresses <LMAn_base> of the LMAn are defined in the first section of this chapter under the key word “Register addresses”.

Register access All registers are accessible in 16-bit width.
Writing to non-existing register bits is ignored, reading of these bits return always 0.

21.5.2 LMAn register details

(1) LMAnCTLL - LMAn control register L

This register enables/disables the LMAn operation.

Access This register can be read or written in 16-bit units.

Address <LMAn_base> + 80_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	LMAn MD[1:0]		LMAn ACSE	LMAn SCHE	LMAn AFE	LMAn ITMK	LMAn IRMK
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-20 LMAnCTLL register contents (1/2)

Bit position	Bit name	Function
6, 5	LMAn MD[1:0]	LMAn operation mode selection **MD[1:0]** / **Operation mode** 00_B: UART through mode. The LMAn is bypassed and the connected UART is used as a normal UART. 01_B: UART buffer mode. The connected UART is used UART with buffer. 10_B: LIN master mode without break in header. The connected UART is used as LIN master. Data transmission is continued even if data consistency errors within the LIN header (BF, SF) are detected. However, in case of an data consistency error in the PID data transmission is stopped. 11_B: LIN master mode with break in header. The connected UART is used as LIN master. Data transmission is stopped if data consistency errors within the LIN header (BF, SF) are detected. In this case the status interrupt signal INTLMAnTIS is generated. Afterwards a new frame transaction with BF can be started.
4	LMAn ACSE	Automatic checksum calculation enable 0: automatic checksum calculation disabled 1: automatic checksum calculation enabled In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0. If automatic checksum calculation is disabled, the • Rx checksum needs to be calculated by software and compared with the received checksum. • Tx checksum need to be calculated by software and to be set to the TX buffer prior starting data transmission. If automatic checksum calculation is enabled, the • Rx checksum is automatically calculated from the received data and compared with the received checksum. • Tx checksum is automatically calculated and set to the TX buffer upon start of a data transmission.

Table 21-20 LMAnCTLL register contents (2/2)

Bit position	Bit name	Function
3	LMAn SCHE	Scheduler enable 0: scheduler disabled 1: scheduler enabled In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0. Before the scheduler is enabled, the scheduler counter, that is connected to LMAn, needs to be started.
2	LMAn AFE	Automatic frame start function enable 0: automatic frame start function disabled 1: automatic frame start function enabled In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0. If the automatic frame start function is disabled, a frame transmission is started by software, when LMAnTCTLL.LMAnTRQ is set to 1. If the automatic frame start function is enabled, a frame transmission is automatically started by the scheduler immediately after the interframe space, if LMAnTCTLL.LMAnTRQ = 1.
1	LMAn ITMK	Transmission interrupt request (INTLMAnTIT) mask 0: INTLMAnTIT not masked (INTLMAnTIT generated) 1: INTLMAnTIT masked (INTLMAnTIT is not generated) In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0.
0	LMAn IRMK	Reception interrupt request (INTLMAnTIR) mask 0: INTLMAnTIR not masked (INTLMAnTIR generated) 1: INTLMAnTIR masked (INTLMAnTIR is not generated) In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0.

(2) LMAnCTLH - LMAn control register H

This register enables/disables the LMAn operation.

Access This register can be read or written in 16-bit units.

Address <LMAn_base> + 84_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LMAn PW	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-21 LMAnCTLH register contents

Bit position	Bit name	Function
15	LMAn PW	LMAn enable 0: LMAn disabled (clock supply stopped) 1: LMAn enabled (clock supply operating) When LMAnPW is set to 0, all operations are stopped and LMAn is reset.

(3) LMAnSTRL - LMAn status register L

This register provides Rx process status indications.

Access This register can be read in 16-bit units.

Address <LMAn_base> + 88_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LMAn SSB[2:0]			0	LMAn RXBE[11:00]											
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 21-22 LMAnSTRL register contents

Bit position	Bit name	Function
15 to 13	LMAn SSB[2:0]	Rx buffer mode status flags **SSB[2:0]** / **Rx buffer mode** 0_H — Idle state (no data was received) 5_H — Data has been received, but the Rx data length has not been reached. An Rx abort is necessary by LMAnRCTLH.LMAnRAB = 1. other — Abnormal operation has occurred. An Rx abort is necessary by LMAnRCTLH.LMAnRAB = 1. LMAnSSB[2:0] can be read for diagnostic purposes when a Rx process stalls.
11 to 0	LMAn RXBE[11:00]	Rx buffer error flag 0: no UART error detections of data in the Rx buffer 11 to 00 1: UART has detected an error of data in the Rx buffer 11 to 00 The bit number [11:00] corresponds to the Rx buffer number: 0: Rx byte 0 (LMAnRX01.LMAnRX00B[7:0] caused an error ... 11: Rx byte (LMAnRXAB.LMAnRX11B[7:0] caused an error Each bit remains at 1 until it is cleared by LMAnSTCL.LMAnCLRRXBE[11:00] = 1.

(4) LMAnSTRH - LMAn status register H

This register provides Rx process status indications.

Access This register can be read in 16-bit units.

Address <LMAn_base> + $8C_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LMAn SST	LMAn SSR	LMAn TXEF	LMAn RXFF	LMAn ROVE	LMAn FCSE	LMAn SRF	LMAn PIE	LMAn BFE	LMAn SFE	LMAn SSL[5:0]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 21-23 LMAnSTRH register contents (1/3)

Bit position	Bit name	Function
15	LMAnSST	Tx status flag 0: no Tx request issued 1: Tx request issued LMAnSST is automatically set if a Tx request was set by LMAnTCTLL.LMAnTRQ= 1. LMAnSST is automatically set when a new frame automatically starts if the automatic frame start function is used in LIN mode (LMAnCTLL.LMAnAFE = 1). This flag is not set in UART through mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • Tx process completion (frame transmission completed in LIN master mode)
14	LMAnSSR	Rx status flag 0: no Rx request issued 1: Rx request issued LMAnSSR is automatically set if a Rx request was set by LMAnRCTLL.LMAnRRQ= 1. This flag is not set in UART through or LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • Rx process completion
13	LMAnTXEF[a]	Tx buffer empty flag 0: data remaining in Tx buffer to be transmitted 1: Tx buffer empty: last Tx data transmitted This flag is not set in UART through mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLTXEF = 1

Table 21-23 LMAnSTRH register contents (2/3)

Bit position	Bit name	Function
12	LMAnRXFF[a]	Rx buffer full flag 0: data remaining to be received and to be stored in Rx buffer 1: Rx buffer full: last Rx data received This flag is not set in UART through mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLRXFF = 1
11	LMAnROVE[a]	Rx buffer overflow flag 0: no Rx buffer overflow occurred 1: Rx buffer overflow occurred When an overflow occurs during data reception, the new data is not stored but discarded. This flag is not set in UART through or LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLROVE = 1
10	LMAnFCSE[a]	Checksum error flag 0: no checksum error occurred 1: checksum error occurred LMAnFCSE indicates the result of the checksum control during a LIN frame reception. An error is indicated, if the checksum, calculated from the received data, does not match the received checksum. This flag is only effective in LIN master mode with automatic checksum function enabled (LMAnCTLL.LMAnACSE = 1). It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLFCSE = 1
9	LMAnSRF[a]	Scheduler ready flag 0: no scheduler ready event occurred 1: scheduler ready event occurred A scheduler ready event occurs, when the value of the compare register LMAnCMPL.LMAnCMP[15:0] matches the value of the scheduler counter. In that case a status interrupt request INTLMAnTIS is also asserted. This flag is only effective in LIN master mode with scheduler function enabled (LMAnCTLL.LMAnSCHE = 1), while automatic frame start function remains disabled (LMAnCTLL.LMAnAFE = 0). It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLSRF = 1

Table 21-23 LMAnSTRH register contents (3/3)

Bit position	Bit name	Function
8	LMAnPIE[a]	Preparation incomplete error flag 0: Rx/Tx buffer preparation correct 1: Rx/Tx buffer preparation incorrect LMAnPIE is set, if a LIN master frame transfer is started, though the Rx respectively Tx buffer has not been prepared correctly. An incorrect buffer preparation is detected upon any of the following conditions: • LMAnTXEF = 1 (Tx buffer empty) • LMAnSTRH.LMAnRXFF = 1 (Rx buffer full) • LMAnSTRH.LMAnTCTLL.LMAnTLG[3:0] holds incorrect value This flag is only effective in LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLPIE = 1
7	LMAnBFE[a]	BF (Break Field) error flag 0: BF transmission successful 1: BF transmission failed LMAnBFE is set, if a framing, overrun or data consistency check error was detected during BF transmission at start of a LIN frame transmission. This flag is only effective in LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLBFE = 1
6	LMAnSFE[a]	SF (Sync Field) error flag 0: SF transmission successful 1: SF transmission failed LMAnSFE is set, if a framing, overrun or data consistency check error was detected during SF transmission at start of a LIN frame transmission. In case of an error detection the status interrupt request INTLMAnTIS is asserted. This flag is only effective in LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLSFE = 1
5 to 0	LMAn SSL[5:0]	LIN master mode status flag **LMAnSSL[5:0]** / **LIN master mode status** 0_H / Idle state (no operation) 19_H / No response from slave. An Tx abort is necessary by LMAnRCTLH.LMAnTAB = 1. other / Abnormal operation has occurred. An Tx abort is necessary by LMAnRCTLH.LMAnTAB = 1. LMAnSSL[2:0] can be read for diagnostic purposes when a LIN master transmission process stalls.

a) Before starting a data transmission process, these flags should be cleared by setting the corresponding status clear bit in LMAnSTC register to 1.

(5) LMAnSTCL - LMAn status clear register L

This register is used to clear the status and error bits of the LMAn status register L LMAnSTRL.

Access This register can be written in 16-bit units.
Reading this register returns an undefined value.

Address <LMAn_base> + 90_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	LMAn CLRXBE[11:00]											
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 21-24 LMAnSTCL register contents

Bit position	Bit name	Function
11 to 0	LMAn CLRXBE [11:00]	Clear Rx buffer error flags 0: writing 0 is ignored 1: writing 1 clears LMAnRXBE[11:00]

(6) LMAnSTCH - LMAn status clear register H

This register is used to clear the status and error bits of the LMAn status register L LMAnSTRH.

Access This register can be written in 16-bit units.
Reading this register returns an undefined value.

Address <LMAn_base> + 94_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	LMAnCLTXEF	LMAnCLRXFF	LMAnCLROVE	LMAnCLFCSE	LMAnCLSRF	LMAnCLPIE	LMAnCLBFE	LMAnCLSFE	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 21-25 LMAnSTCH register contents

Bit position	Bit name	Function
13	LMAn CL TXEF	Tx buffer empty flag 0: writing 0 is ignored 1: writing 1 clears LMAnTXEF
12	LMAn CL RXFF	Rx buffer full flag 0: writing 0 is ignored 1: writing 1 clears LMAnRXFF
11	LMAn CL ROVE	Rx buffer overflow flag 0: writing 0 is ignored 1: writing 1 clears LMAnROVE
10	LMAn CL FCSE	Checksum error flag 0: writing 0 is ignored 1: writing 1 clears LMAnFCSE
9	LMAn CL SRF	Scheduler ready flag 0: writing 0 is ignored 1: writing 1 clears LMAnSRF
8	LMAn CL PIE	Preparation incomplete error flag 0: writing 0 is ignored 1: writing 1 clears LMAnPIE
7	LMAn CL BFE	BF (Break Field) error flag 0: writing 0 is ignored 1: writing 1 clears LMAnBFE
6	LMAn CL SFE	SF (Sync Field) error flag 0: writing 0 is ignored 1: writing 1 clears LMAnSFE

(7) LMAnCMPL - LMAn compare register L

This register holds the scheduler comparison value.

Access This register can be read in 16-bit units.

Address <LMAn_base> + 98_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LMAnCMP[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 21-26 LMAnCMPL register contents

Bit position	Bit name	Function
15 to 0	LMAn CMP[15:00]	Current scheduler comparison value LMAnCMP[15:00] is loaded with the sum of the current free-running counter value CNATmCNT[15:0] and the frame slot length FRSL upon start of a LIN master frame.

(8) LMAnTCTL - LMAn Tx control register L

This register controls the LMAn Tx buffer.

Access This register can be read or written in 16-bit units.

Address <LMAn_base> + $D8_H$

Initial Value 0000_H. This register is initialized by any reset and by LMAnCTLH.LMAnPW = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	LMAn SLEC	LMAn SLRT	LMAn FRQ	LMAn TRQ	LMAn TLG[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-27 LMAnTCTL register contents (1/2)

Bit position	Bit name	Function
7	LMAn SLEC	Checksum control 0: classic checksum Only the data bytes are used for checksum calculation. 1: enhanced checksum Checksum is calculated from the data and the PID. LMAnSLEC is only effective, if automatic checksum calculation is enabled (LMAnCTLL.LMAnACSE = 1).
6	LMAn SLRT	LIN master mode operation control 0: Tx mode 1: Rx mode LMAnSLRT is only effective in LIN master mode.
5	LMAn FRQ	First Tx request control 0: LIN frame transaction starts with the scheduler ready event 1: LIN frame transaction starts immediately by requesting transmission (LMAnTRQ = 1), in case no Tx operation has been requested (LMAnSTRH.LMAnSST = 0). In case LMAnFRQ is set to 1 while LMAnSTRH.LMAnSST = 1, LIN frame transaction starts with the next scheduler event, and thus behaves as with LMAnFRQ = 0. Set LMAnFRQ = 1 together with LMAnTRQ = 1. LMAnFRQ is only effective in LIN master mode with using the scheduler (LMAnCTLL/LMAnSCHE = 1) and auto frame start enabled (LMAnCTLL.LMAnAFE = 1). In all other modes, LMAnFRQ shall be set to 0.
4	LMAn TRQ	Tx request control 0: Tx operation has started or has not been requested. 1: Tx operation request In LIN master mode LMAnTRQ = 1 triggers LIN frame transaction also in LIN master Rx mode (LMAnSLRT = 1. After setting LMAnTRQ = 1, LMAnTRQ returns automatically to 0 when transmission has started. Writing 0 to LMAnTRQ has no effect. In UART through (LMAnMD[1:0] = 00_B) this bit must be set to 0. **Caution:** It is prohibited to set LMAnTRQ = 1 during a pending transmission request (LMAnSTRH.LMAnSST = 1), except LMAn is operating in LIN master mode with scheduler and auto frame start (LMAnCTLL register bits LMAnSCHE = LMAnAFE = 1).

Table 21-27 LMAnTCTL register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>3 to 0</td><td>LMAn TLG[3:0]</td><td>Tx buffer length specification
LMAnTLG[3:0] are only effective in UART buffer and LIN master mode.
• in UART buffer mode
<table>
<tr><th>LMAnTLG[3:0]</th><th>Tx buffer length</th></tr>
<tr><td>0</td><td>12 byte transmission</td></tr>
<tr><td>1 to 12</td><td>1 to 12 byte transmission in UART buffer mode</td></tr>
<tr><td>13 to 15</td><td>prohibited</td></tr>
</table>
• in LIN master mode
<table>
<tr><th>LMAnTLG[3:0]</th><th>Tx buffer length</th></tr>
<tr><td>0 to 1</td><td>prohibited</td></tr>
<tr><td>2 to 10</td><td>2 to 10 byte transmission in LIN master mode</td></tr>
<tr><td>11 to 15</td><td>prohibited</td></tr>
</table>
The value set to LMAnTLG[3:0] includes the PID and checksum bytes, thus a maximum of 8 data byte can be transmitted.
If any of the prohibited values are set to LMAnTLG[3:0], a preparation incomplete error is detected and indicated by LMAnSTRH.LMAnPIE = 1.</td></tr>
</table>

(9) LMAnTCTH - LMAn Tx control register H

This register controls the Tx abort process.

Access This register can be read or written in 16-bit units.

Address <LMAn_base> + DC_H

Initial Value 0000_H. This register is initialized by any reset and by LMAnCTLH.LMAnPW = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LMAn TAB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-28 LMAnTCTH register contents

Bit position	Bit name	Function
0	LMAn TAB	Tx abort request 0: reading returns always 0, writing 0 has no effect 1: writing 1 requests Tx abort If LMAnTAB is set to 1, Tx operation is stopped accordingly and the Tx status flag LMAnSTRH.LMAnSST is cleared. This bit has no effect in UART through mode. This bit is also effective during Rx process in LIN master mode.

Note If Tx is aborted by LMAnTAB = 1, the LMAn stops to send any further data to the UART. However ongoing transmissions are completed by the UART. The complete stop of transmissions can be confirmed by URTEnSTR0.URTEnSST = 0.

(10) LMAnRCTL - LMAn Rx control register L

This register controls the LMAn Rx buffer.

Access This register can be read or written in 16-bit units.

Address <LMAn_base> + $F8_H$

Initial Value 0000_H. This register is initialized by any reset and by LMAnCTLH.LMAnPW = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	LMAn RRQ	LMAn RLG[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-29 LMAnRCTL register contents

Bit position	Bit name	Function
4	LMAn RRQ	Rx request control 0: Rx operation has started or has not been requested. 1: Rx operation request After setting LMAnRRQ = 1, LMAnRRQ returns automatically to 0 when reception data storage to the Rx buffer has started. Writing 0 to LMAnRRQ has no effect. This bit is only effective in UART buffer mode. In all other modes this bit must not be set to 1.
3 to 0	LMAn RLG[3:0]	Rx buffer length specification LMAnRLG[3:0] are only effective in UART buffer mode. **LMAnRLG[3:0] / Rx buffer length** 0 / continuous data reception 1 to 12 / 1 to 12 byte reception to Rx buffer 13 to 15 / prohibited In continuous data reception (LMAnRLG[3:0] = 0) received data is stored to the Rx buffer continuously. After each 12 byte storage a reception interrupt INTLMAnTIR or status interrupt INTLMAnTIS request is generated. To stop continuous storage set LMAnRLG[3:0] to 1 to 12.

(11) LMAnRCTH - LMAn Rx control register H

This register controls the Rx abort process.

Access This register can be read or written in 16-bit units.

Address <LMAn_base> + FC_H

Initial Value 0000_H. This register is initialized by any reset and by LMAnCTLH.LMAnPW = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LMAn RAB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-30 LMAnRCTH register contents

Bit position	Bit name	Function
0	LMAn RAB	Rx abort request 0: reading returns always 0, writing 0 has no effect 1: writing 1 requests Rx abort If LMAnRAB is set to 1, Rx operation is stopped accordingly and the Rx status flag LMAnSTRH.LMAnSSR is cleared. This bit is only effective in UART buffer mode. In all other modes this bit must not be set to 1. This bit is also effective during Rx process in LIN master mode.

Note If Rx is aborted by LMAnRAB = 1, the LMAn stops to store any data in the Rx buffer. However ongoing reception by the UART will be completed.
The complete stop of reception can be confirmed by URTEnSTR0.URTEnSSR = 0.

Chapter 22 CAN Controller (FCN)

The microcontroller features on-chip CAN (Controller Area Network) controllers that complies with the CAN protocol as standardized in ISO 11898.

This chapter describes the CAN controller (FCN).

The first section describes all V850E2/Sx4-H specific properties, such as instances, register base addresses, and input/output signal names.
The subsequent sections describe the features that apply to all implementations.

22.1 V850E2/Sx4-H FCN Features

Instances

This microcontroller has the following number of instances of FCN:

Table 22-1 Instances of FCN

FCN	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of instances	1	2	2
Name	FCN0	FCN0, FCN1	FCN0, FCN1

Instances index n

Throughout this chapter, the instance of an FCN is identified by the index “n” (n = 0 or 1), for example, FCNnGMCLCTL for the FCNn global control register.

Table 22-2 Message buffers of FCN

Channel	Number of message buffers
FCN0	64
FCN1	64

Message buffers index m

Throughout this chapter, the FCN message buffer registers are identified by “m” (m = 0 to 63), for example, FCNnMmDAT4B for FCN instance n, message data byte 4 of FCN message buffer register m.

Register addresses

All FCN register addresses are given as addresses offset from the individual base address <FCNn_base>.
The <FCNn_base> address of each FCNn is given in the following table:

Table 22-3 Register base addresses <FCNn_base>

FCNn	<FCNn_base> address
FCN0	FF48 0000_H
FCN1	FF4A 0000_H

Clock supply The following clocks are supplied to FCN:

Table 22-4 FCNn clock supply

FCNn	Clock	Connected to:
FCN0	PCLK	Clock generator CKSCLK_113
FCN1	PCLK	Clock generator CKSCLK_113

Interrupts FCN can generate the following interrupt requests:

Table 22-5 FCNn interrupt requests

FCNn signals	Function	Connected to:
FCN0:		
INTC0ERR	Error indication	INTFCN0ERR
INTC0REC	Reception completion	INTFCN0REC
INTC0TRX	Transmission completion	INTFCN0TRX
INTC0WUP	Sleep wake-up/ transmission abortion	INTFCNWUP
FCN1:		
INTC1ERR	Error indication	INTFCN1ERR
INTC1REC	Reception completion	INTFCN1REC
INTC1TRX	Transmission completion	INTFCN1TRX
INTC0WUP	Sleep wake-up/ transmission abortion	INTFCNWUP

FCN hardware reset FCN and its registers are initialized by the following reset signal:

Table 22-6 FCNn reset signal

FCNn	Reset signal
FCN0	System reset SYSRES

I/O signals The I/O signals of FCN are listed in the table below.

Table 22-7 FCNn I/O signals

FCNn signals	Function	Connected to:
FCN0:		
CRXD0	CAN bus receive input	Ports FCN0RX and FCN1RX
CTXD0	CAN bus transmit output	Port FCN0TX
FCN1:		
CRXD1	CAN bus receive input	Port FCN1RX
CTXD1	CAN bus transmit output	Port FCN1TX

Time stamp The following FCNn time stamp output signals can be internally connected to a capture input of TAUA.

Table 22-8 FCNn time stamp signals

FCNn signals	Function	Connected to:
FCN0:		
TSOUT	CAN time stamp output	TAUA0 TAUA0TTIN0
FCN1:		
TSOUT	CAN time stamp output	TAUA0 TAUA1TTIN1

<R>

Note For details, see *15.2 “TAUA Input Selection”*.

22.2 FCN0 and FCN1 Connection

The FCN0 and FCN1 modules have the option to be connected to the same CAN bus signals. This allow to operate the two FCN modules on the same CAN bus (FCN1 signals) thus allowing 128 message buffer support on this bus.

The following figure depicts the FCN0 and FCN1 connection scheme:

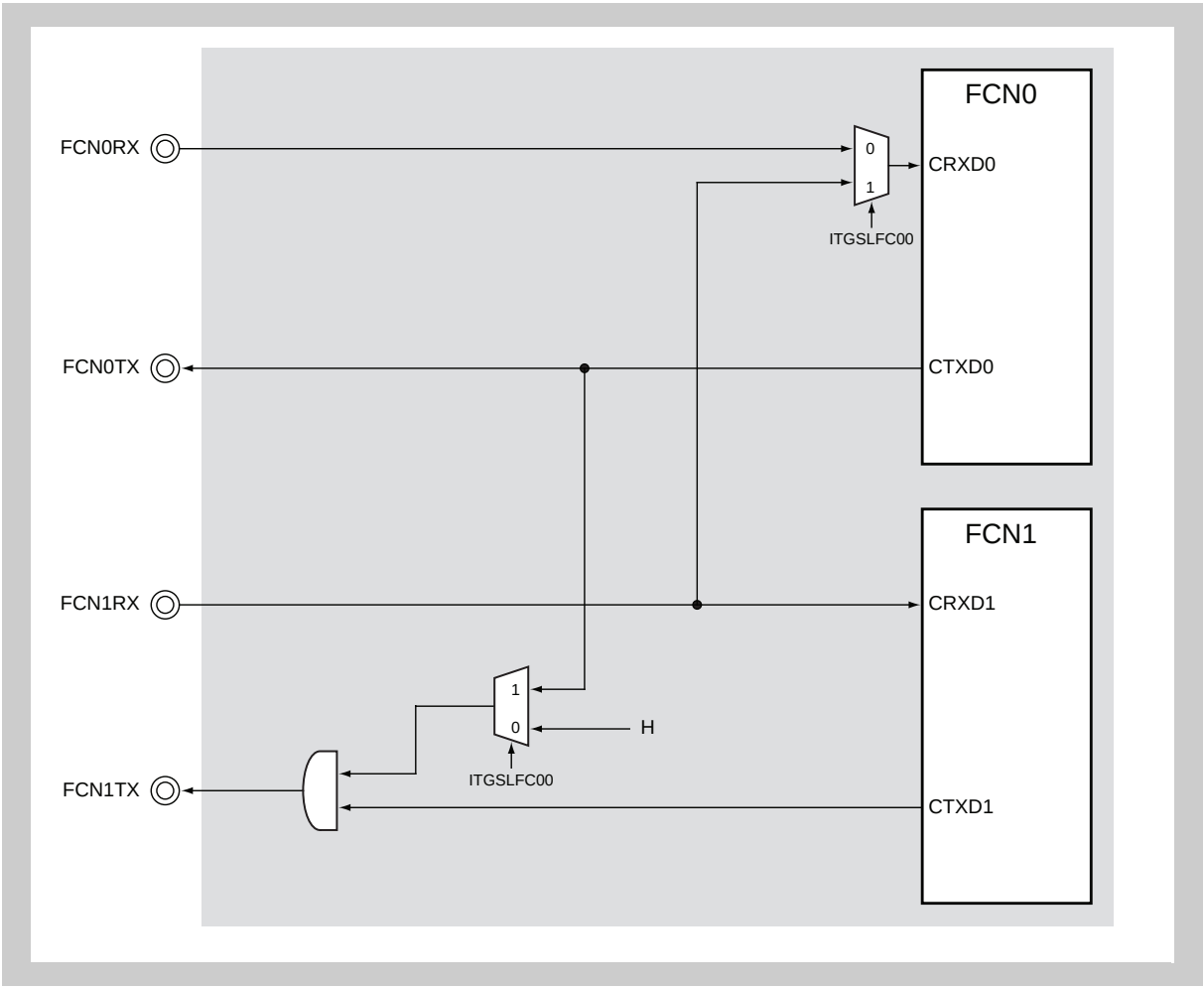


Figure 22-1 FCN0 and FCN1 connection scheme

(1) ITGSLFC0 - FCN0 signal connection selection register

This register selects the signals of FCN0.

Access This register can be read/written in 8-bit units.

Address FF77 2008$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ITGSLFC00
R	R	R	R	R	R	R	R/W

Table 22-9 ITGSLFC0 register contents

Bit position	Bit name	Function
0	ITGSLFC00	FCN0 signal selection 0: use FCN0 bus signals (FCN0RX, FCN0TX) 1: use FNC1 bus signals (FCN1RX, FCN1TX) - combined operation

22.3 Features

- Compliant with ISO 11898 and tested according to ISO/DIS 16845 (CAN conformance test)
- Standard frame and extended frame transmission/reception enabled
- Transfer rate: 1 Mbps max. (If FCN clock input is ≥ 16 MHz)
- 64 or 128 message buffers per channel
- Receive/transmit history list function (can be set individually for each message buffer)
- Automatic block transmission function
- Multi-buffer receive block function
- Mask setting of 8 patterns is possible for each channel, applicable for data and remote frames
- Data bit time, communication baud rate and sample point can be controlled by FCN module bit-rate prescaler register (FCNnCMBRPRS) and bit rate register (FCNnCMBTCTL)
  - As an example the following sample-point configurations can be configured:
    66.7%, 70.0%, 75.0%, 80.0%, 81.3%, 85.0%, 87.5%
  - Baud rates in the range of 10 kbps up to 1 Mbps can be configured
- Enhanced features:
  - Each message buffer can be configured to operate as a transmit or a receive message buffer
  - A transmission request can be aborted by clearing the Transmit-Request flag of the concerned message buffer. Supported by Transmission Abort Interrupt, on successful abortion.
  - Automatic block transmission operation mode (ABT)
  - Time stamp function for FCN channels 0 and 1 in collaboration with timers capture channels
  - A centralized global data update bit monitor register makes it possible to check all data update bits from one location

22.3.1 Overview of functions

Table 22-10 "Overview of functions" presents an overview of the CAN Controller functions.

Table 22-10 Overview of functions

Function	Details
Protocol	CAN protocol ISO 11898 (standard and extended frame transmission/reception)
Baud rate	Maximum 1 Mbps (minimum FCN clock input = 16 MHz)
Data storage	Storing messages in the FCN RAM
Number of messages	• 64/128 message buffers per channel • Each message buffer can be set to be either a transmit message buffer or a receive message buffer.
Message reception	• Unique ID can be set to each message buffer. • Mask setting of 8 patterns is possible for each channel, applicable for data and remote frames • A receive completion interrupt is generated each time a message is received and stored in a message buffer (receive completion interrupts can be enabled/disabled for each message buffer) • Two or more receive message buffers can be used as a FIFO receive buffer (multi-buffer receive block function). • Receive history list function (can be set individually for each message buffer) • Centralized global data update bit monitor register
Message transmission	• Unique ID can be set to each message buffer. • Receive completion interrupts can be enabled/disabled for each message buffer • Transmit Abort interrupt and transmission completely finished flag for each message buffer (only one transmission of any buffer can be aborted at a time) • Message buffer numbers 0 to 15/31 specified as the transmit message buffers can be used for automatic block transfer. The message transmission interval is programmable (using the automatic block transmission ("ABT") function). • Transmission history list function (can be set individually for each message buffer)
Remote frame processing	• Remote frame processing by transmit message buffer • Remote frame processing by receive message buffer, when applying one of the 8 masks
Time stamp function	• The time stamp function can be set for a message reception when a 16-bit timer is used in combination. • Time stamp capture trigger can be selected (SOF or EOF in a CAN message frame can be detected.).
Diagnostic function	• Readable error counters • "Valid protocol operation flag" for verification of bus connections • Receive-only mode • Single-shot mode • CAN protocol error identification • Self-test mode
Release from bus-off state	• Forced release from bus-off possible by software. • No automatic release from bus-off (software must send recovery request).
Power save mode	• CAN sleep mode (can be woken up by CAN bus) • CAN stop mode (cannot be woken up by CAN bus)

22.3.2 Configuration

The CAN Controller is composed of the following four blocks.

- APB interface
  This functional block provides an APB interface and a means of transmitting and receiving messages between the FCN module and the host CPU.
- MCM (Message Control Module)
  This functional block controls access to the CAN protocol layer and to the FCN RAM within the FCN module.
- CAN protocol layer
  This functional block is involved in the operation of the CAN protocol and its related settings.
- CAN RAM
  This is the CAN memory functional block, which is used to store message IDs, message data, etc.

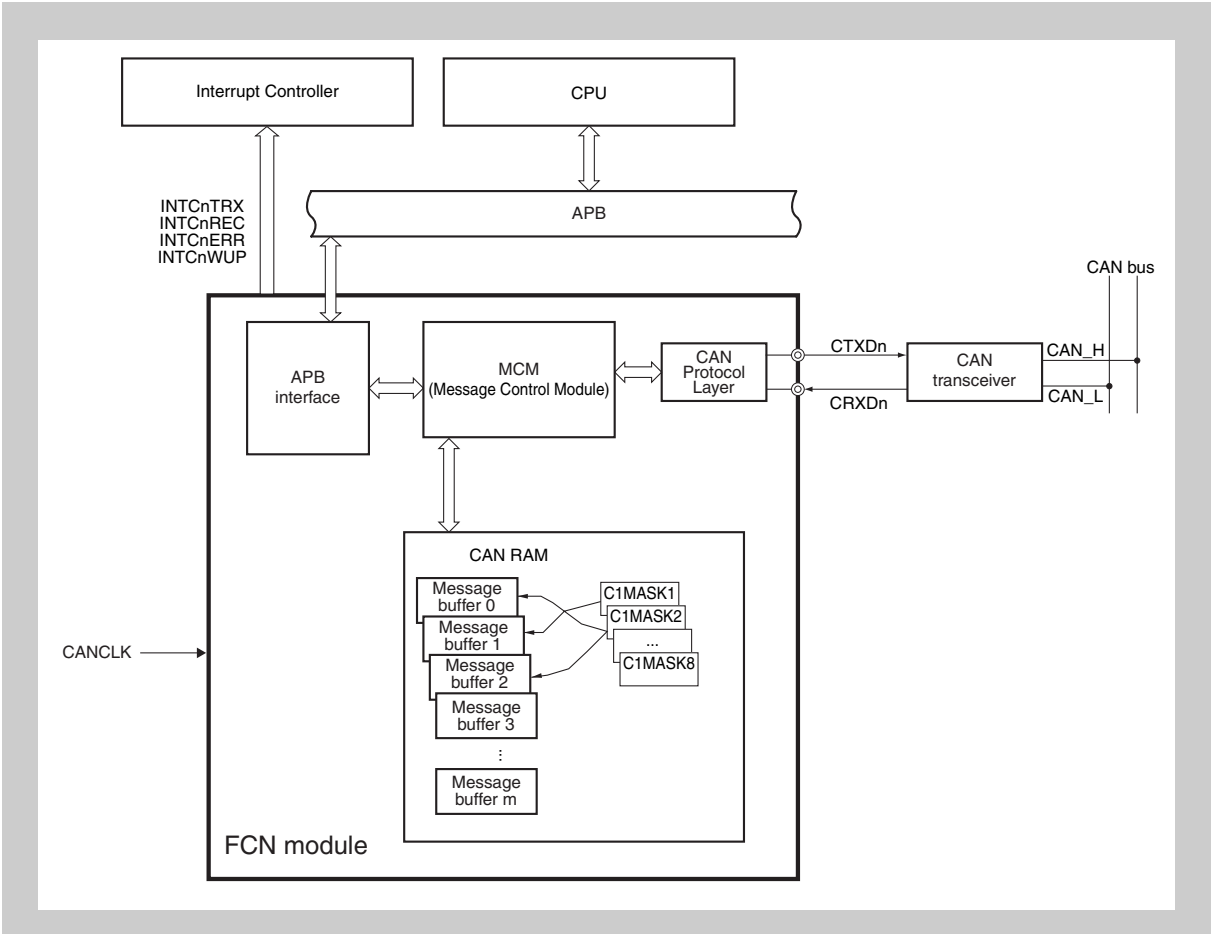


Figure 22-2 Block diagram of the CAN Controller

<R> **Caution** The CAN RAM includes a memory reference module. If a CAN RAM error occurs during a software reset, the message buffer RAM read error detection bit (FCNnGMCLCTL.FCNnGMCLECCF) will be set. If this bit is set, be sure to check the FCN function for errors.

22.4 Internal Registers of FCN

22.4.1 CAN Controller configuration

Table 22-11 List of FCN registers (1/2)

Item	Register Name
FCNn global registers	FCNn global control register (FCNnGMCLCTL)
	FCNn global clock selection register (FCNnGMCSPRE)
	FCNn global automatic block transmission control register (FCNnGMABCTL)
	FCNn global automatic block transmission delay setting register (FCNnGMADCTL)
	FCNn global Data New bit monitor registers (FCNnDNBMRX0 - FCNnDNBMRX3)
FCNn module registers	FCNn module mask 1 registers (FCNnCMMKCTL01H, FCNnCMMKCTL02H, FCNnCMMKCTL01W)
	FCNn module mask 2 registers (FCNnCMMKCTL03H, FCNnCMMKCTL04H, FCNnCMMKCTL03W)
	FCNn module mask3 registers (FCNnCMMKCTL05H, FCNnCMMKCTL06H, FCNnCMMKCTL05W)
	FCNn module mask 4 registers (FCNnCMMKCTL07H, FCNnCMMKCTL08H, FCNnCMMKCTL07W)
	FCNn module mask 5 registers (FCNnCMMKCTL09H, FCNnCMMKCTL10H, FCNnCMMKCTL09W)
	FCNn module mask 6 registers (FCNnCMMKCTL11H, FCNnCMMKCTL12H, FCNnCMMKCTL11W)
	FCNn module mask 7 registers (FCNnCMMKCTL13H, FCNnCMMKCTL14H, FCNnCMMKCTL13W)
	FCNn module mask 8 registers (FCNnCMMKCTL15H, FCNnCMMKCTL16H, FCNnCMMKCTL15W)
	FCNn module control register (FCNnCMCLCTL)
	FCNn module last error information register (FCNnCMLCSTR)
	FCNn module information register (FCNnCMINCSTR)
	FCNn module error counter register (FCNnCMERCNT)
	FCNn module interrupt enable register (FCNnCMIECTL)
	FCNn module interrupt status register (FCNnCMISCTL)
	FCNn module bit rate prescaler and FCN clock selector register (FCNnCMBRPRS)
	FCNn module bit rate register (FCNnCMBTCTL)
	FCNn module last in-pointer register (FCNnCMLISTR)
	FCNn module receive history list register (FCNnCMRGRX)
	FCNn module last out-pointer register (FCNnCMLOSTR)
	FCNn module transmit history list register (FCNnCMTGTX)
	FCNn module time stamp register (FCNnCMTSCTL)

Table 22-11 List of FCN registers (2/2)

Item	Register Name
FCNn message buffer registers	FCNn message data byte 0 to 3 registers m (FCNnMmDAT0W, FCNnMmDAT0H, FCNnMmDAT2H, FCNnMmDAT0B, FCNnMmDAT1B, FCNnMmDAT2B, FCNnMmDAT3B)
	FCNn message data byte 4 to 7 registers m (FCNnMmDAT4W, FCNnMmDAT4H, FCNnMmDAT6H, FCNnMmDAT4B, FCNnMmDAT5B, FCNnMmDAT6B, FCNnMmDAT7B)
	FCNn message data length register m (FCNnMmDTLGB)
	FCNn message configuration register m (FCNnMmSTRB)
	FCNn message ID registers m (FCNnMmMID0H, FCNnMmMID1H, FCNnMmMID0W)
	FCNn message control register m (FCNnMmCTL)

22.4.2 CAN Controller Registers Overview

Address offset All register addresses are given as offsets to the base address <FCNn_base>. The <FCNn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

(1) FCNn global and module registers

Table 22-12 FCNn global and module registers (1/2)

Address offset	Register name	Symbol	R/W	Access bit	After reset
0 0008$_H$	FCNn global clock selection register	FCNnGMCSPRE	R/W	8	0F$_H$
0 0020$_H$	FCNn global automatic block transmission delay setting register	FCNnGMADCTL	R/W	8	00$_H$
0 8000$_H$	FCNn global control register	FCNnGMCLCTL	R/W	16	00x0$_H$[a]
0 8018$_H$	FCNn global automatic block transmission control register	FCNnGMABCTL	R/W	16	0000$_H$
1 00C0$_H$	FCNn global data update bit monitor register 0	FCNnDNBMRX0	R	32	[b]
1 00D0$_H$	FCNn global data update bit monitor register 1	FCNnDNBMRX1	R	32	[b]
1 00E0$_H$	FCNn global data update bit monitor register 2	FCNnDNBMRX2	R	32	[b]
1 00F0$_H$	FCNn global data update bit monitor register 3	FCNnDNBMRX3	R	32	[b]
0 8300$_H$	FCNn module mask 1 register	FCNnCMMKCTL01H	R/W	16	[b]
0 8308$_H$		FCNnCMMKCTL02H			
1 0300$_H$		FCNnCMMKCTL01W		32	
0 8310$_H$	FCNn module mask 2 register	FCNnCMMKCTL03H	R/W	16	[b]
0 8318$_H$		FCNnCMMKCTL04H			
1 0310$_H$		FCNnCMMKCTL03W		32	
0 8320$_H$	FCNn module mask 3 register	FCNnCMMKCTL05H	R/W	16	[b]
0 8328$_H$		FCNnCMMKCTL06H			
1 0320$_H$		FCNnCMMKCTL05W		32	
0 8330$_H$	FCNn module mask 4 register	FCNnCMMKCTL07H	R/W	16	[b]
0 8338$_H$		FCNnCMMKCTL08H			
1 0330$_H$		FCNnCMMKCTL07W		32	
0 8340$_H$	FCNn module mask 5 register	FCNnCMMKCTL09H	R/W	16	[b]
0 8348$_H$		FCNnCMMKCTL10H			
1 0340$_H$		FCNnCMMKCTL09W		32	
0 8350$_H$	FCNn module mask 6 register	FCNnCMMKCTL11H	R/W	16	[b]
0 8358$_H$		FCNnCMMKCTL12H			
1 0350$_H$		FCNnCMMKCTL11W		32	
0 8360$_H$	FCNn module mask 7 register	FCNnCMMKCTL13H	R/W	16	[b]
0 8368$_H$		FCNnCMMKCTL14H			
1 0360$_H$		FCNnCMMKCTL13W		32	
0 8370$_H$	FCNn module mask 8 register	FCNnCMMKCTL15H	R/W	16	[b]
0 8378$_H$		FCNnCMMKCTL16H			
1 0370$_H$		FCNnCMMKCTL15W		32	
0 0248$_H$	FCNn module last error information register	FCNnCMLCSTR	R/W	8	00$_H$

Table 22-12 FCNn global and module registers (2/2)

Address offset	Register name	Symbol	R/W	Access bit	After reset
0 024C$_H$	FCNn module information register	FCNnCMINSTR	R	8	00$_H$
0 0268$_H$	FCNn module bit-rate prescaler and clock selector register	FCNnCMBRPRS	R/W	8	FF$_H$
0 0278$_H$	FCNn module last receive pointer register	FCNnCMLISTR	R	8	Undefined
0 0288$_H$	FCNn module last transmit pointer register	FCNnCMLOSTR	R	8	Undefined
0 8240$_H$	FCNn module control register	FCNnCMCLCTL	R/W	16	0000$_H$
0 8250$_H$	FCNn module error counter register	FCNnCMERCNT	R	16	0000$_H$
0 8258$_H$	FCNn module interrupt enable register	FCNnCMIECTL	R/W	16	0000$_H$
0 8260$_H$	FCNn module interrupt status register	FCNnCMISCTL	R/W	16	0000$_H$
0 8270$_H$	FCNn module bit-rate register	FCNnCMBTCTL	R/W	16	370F$_H$
0 8280$_H$	FCNn module receive history list register	FCNnCMRGRX	R/W	16	xx02$_H$
0 8290$_H$	FCNn module transmit history list register	FCNnCMTGTX	R/W	16	xx02$_H$
0 8298$_H$	FCNn module time stamp register	FCNnCMTSCTL	R/W	16	0000$_H$

a) Initial value depends on FCNnGMCLCTL.FCNnGMCLECCF, which indicates error detections when reading from message buffer RAM. Refer to the detailed description of the FCNnGMCLCTL register.

b) After resetting, the value will be 0000$_H$ or 00000000$_H$.

22.4.3 Register bit configuration

Table 22-13 FCN global register bit configuration

Address offset	Symbol	Bit 7/ 15/31/ 23	Bit 6/ 14/30/ 22	Bit 5/ 13/29/ 21	Bit 4/ 12/28/ 20	Bit 3/ 11/27/ 19	Bit 2/ 10/26/ 18	Bit 1/9/ 25/17	Bit 0/8/ 24/16
0 8000$_H$	FCNnGMCLCTL (W)	0	0	FCNnGM CLCLMB		0	0	0	FCNnGMC LCLOM
		0	0	0	FCNnGM CLSESR	0	0	FCNnGM CLSEDE	FCNnGMC LSEOM
	FCNnGMCLCTL (R)	0	0	FCNnGM CLECCF	FCNnGM CLSORF	0	0	FCNnGM CLESDE	FCNnGMC LPWOM
		FCNnGM CLSSMO	0	0	0	0	0	0	0
0 0008$_H$	FCNnGMCSPRE	0	0	0	0	FCNnGMCSPRSC[3:0]			
0 8018$_H$	FCNnGMABCTL (W)	0	0	0	0	0	0	0	FCNnGMA BCLAT
		0	0	0	0	0	0	FCNnGM ABSEAC	FCNnGMA BSEAT
	FCNnGMABCTL (R)	0	0	0	0	0	0	FCNnGM ABCLRF	FCNnGMA BABTT
		0	0	0	0	0	0	0	0
0 0020$_H$	FCNnGMADCTL	0	0	0	0	FCNnGMADSSAD[3:0]			
1 00C0$_H$	FCNnDNBMRX0 (R)	FCNnDNBMSSDN[7:0]							
		FCNnDNBMSSDN[15:8]							
		FCNnDNBMSSDN[23:16]							
		FCNnDNBMSSDN[31:24]							
1 00D0$_H$	FCNnDNBMRX1 (R)	FCNnDNBMSSDN[39:32]							
		FCNnDNBMSSDN[47:40]							
		FCNnDNBMSSDN[55:48]							
		FCNnDNBMSSDN[63:56]							
1 00E0$_H$	FCNnDNBMRX2 (R)[a]	FCNnDNBMSSDN[71:64]							
		FCNnDNBMSSDN[79:72]							
		FCNnDNBMSSDN[87:80]							
		FCNnDNBMSSDN[95:88]							
1 00F0$_H$	FCNnDNBMRX3 (R)[a]	FCNnDNBMSSDN[103:96]							
		FCNnDNBMSSDN[111:104]							
		FCNnDNBMSSDN[119:112]							
		FCNnDNBMSSDN[127:120]							

a) Only available with 128 message buffers (m = 0 to 127)

Table 22-14 FCN module mask control 16-bit registers bit configuration

Address offset	Symbol	Bit 15	Bit 14	Bit 13	Bit 12 to 0
0 8300$_H$	FCNnCMMK CTL01H	FCNnCMMKSSID[15:0]			
0 8308$_H$	FCNnCMMK CTL02H	0	0	0	FCNnCMMKSSID[28:16]
0 8310$_H$	FCNnCMMK CTL03H	FCNnCMMKSSID[15:0]			
0 8318$_H$	FCNnCMMK CTL04H	0	0	0	FCNnCMMKSSID[28:16]
0 8320$_H$	FCNnCMMK CTL05H	FCNnCMMKSSID[15:0]			
0 8328$_H$	FCNnCMMK CTL06H	0	0	0	FCNnCMMKSSID[28:16]
0 8330$_H$	FCNnCMMK CTL07H	FCNnCMMKSSID[15:0]			
0 8338$_H$	FCNnCMMK CTL08H	0	0	0	FCNnCMMKSSID[28:16]
0 8340$_H$	FCNnCMMK CTL09H	FCNnCMMKSSID[15:0]			
0 8348$_H$	FCNnCMMK CTL10H	0	0	0	FCNnCMMKSSID[28:16]
0 8350$_H$	FCNnCMMK CTL11H	FCNnCMMKSSID[15:0]			
0 8358$_H$	FCNnCMMK CTL12H	0	0	0	FCNnCMMKSSID[28:16]
0 8360$_H$	FCNnCMMK CTL13H	FCNnCMMKSSID[15:0]			
0 8368$_H$	FCNnCMMK CTL14H	0	0	0	FCNnCMMKSSID[28:16]
0 8370$_H$	FCNnCMMK CTL15H	FCNnCMMKSSID[15:0]			
0 8378$_H$	FCNnCMMK CTL16H	0	0	0	FCNnCMMKSSID[28:16]

Table 22-15 FCN module mask control 32-bit registers bit configuration

Address offset	Symbol	Bit 31	Bit 30	Bit 29	Bit 28 to 0
$1\ 0300_H$	FCNnCMMKCTL01W	0	0	0	FCNnCMMKSSID[28:0]
$1\ 0310_H$	FCNnCMMKCTL03W	0	0	0	FCNnCMMKSSID[28:0]
$1\ 0320_H$	FCNnCMMKCTL05W	0	0	0	FCNnCMMKSSID[28:0]
$1\ 0330_H$	FCNnCMMKCTL07W	0	0	0	FCNnCMMKSSID[28:0]
$1\ 0340_H$	FCNnCMMKCTL09W	0	0	0	FCNnCMMKSSID[28:0]
$1\ 0350_H$	FCNnCMMKCTL11W	0	0	0	FCNnCMMKSSID[28:0]
$1\ 0360_H$	FCNnCMMKCTL13W	0	0	0	FCNnCMMKSSID[28:0]
$1\ 0370_H$	FCNnCMMKCTL15W	0	0	0	FCNnCMMKSSID[28:0]

Table 22-16 FCN module register bit configuration (1/2)

Address offset	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0 8240$_H$	FCNnCM CLCTL (W)	0	FCNnCM CLCLAL	FCNnCM CLCLVL	FCNnCMCLCLPS[1:0]		FCNnCMCLCLOP[2:0]		
		FCNnCM CLSERC	FCNnCM CLCSEAL	0	FCNnCMCLSEPS[1:0]		FCNnCMCSELOP[2:0]		
	FCNnCM CLCTL (R)	FCNnCM CLERCF	FCNnCM CLALBF	FCNnCM CLVALF	FCNnCMCLMDPF[1:0]		FCNnCMCLMDOF[2:0]		
		0	0	0	0	0	0	FCNnCMC LSSRS	FCNnCMC LSSTS
0 00248$_H$	FCNnCM LCSTR (W)	0	0	0	0	0	0	0	0
	FCNnCM LCSTR (R)	0	0	0	0	0	FCNnCMLCSSLC[2:0]		
0 024CH	FCNnCM INSTR	0	0	0	FCNnCMI NBOFF	FCNnCMINSSTE[1:0]		FCNnCMINSSRE[1:0]	
0 8250$_H$	FCNnCM ERCNT	FCNnCMERTECF[7:0]							
		FCNnCM ERRPSF	FCNnCMERRECF[6:0]						
0 8258$_H$	FCNnCM IECTL (W)	0	FCNnCMIECLIE[6:0]						
		0	FCNnCMIESEIE[6:0]						
	FCNnCM IECTL (R)	0	FCNnCMIEINTF[6:0]						
		0	0	0	0	0	0	0	0
0 8260$_H$	FCNnCM ISCTL (W)	0	FCNnCMISCLTS[6:0]						
		0	0	0	0	0	0	0	0
	FCNnCM ISCTL (R)	0	FCNnCMISITSF[6:0]						
		0	0	0	0	0	0	0	0
0 0268$_H$	FCNnCM BRPRS	FCNnCMBRPRS[7:0]							
0 8270$_H$	FCNnCM BTCTL	0	0	0	0	FCNnCMBTS1LG[3:0]			
		0	0	FCNnCMBTJWLG[1:0]		0	FCNnCMBTS2LG[2:0]		
0 0278$_H$	FCNnCM LISTR	FCNnCMLISSLR[7:0]							
0 8280$_H$	FCNnCM RGRX (W)	0	0	0	0	0	0	0	FCNnCMR GCLRV
		0	0	0	0	0	0	0	0
	FCNnCM RGRX (R)	0	0	0	0	0	0	FCNnCMR GSSPM	FCNnCMR GRVFF
		FCNnCMRDSSPT[7:0]							
<R> 0 0288$_H$	FCNnCM LOSTR	FCNnCMLOSSLT[7:0]							
0 8290$_H$	FCNnCM TGTX (W)	0	0	0	0	0	0	0	FCNnCMT GCLTV
		0	0	0	0	0	0	0	0
	FCNnCM TGTX (R)	0	0	0	0	0	0	FCNnCMT GSSPM	FCNnCMT GTVFF
		FCNnCMTGSSPT[7:0]							

Table 22-16 FCN module register bit configuration (2/2)

Address offset	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
$0\ 8298_H$	FCNnCM TSCTL (W)	0	0	0	0	0	FCNnCMT SCLLK	FCNnCMT SCLSL	FCNnCMT SCLTS
		0	0	0	0	0	FCNnCMT SSELK	FCNnCMT SSESL	FCNnCMT SSETS
	FCNnCM TSCTL (R)	0	0	0	0	0	FCNnCMT SLOKE	FCNnCMT SSELE	FCNnCMT STSGE
		0	0	0	0	0	0	0	0

<R> **Table 22-17 FCN message buffer register bit configuration (1/2)**

Address offset	Symbol	Bit 7/15/ 31/23	Bit 6/14/ 30/22	Bit 5/13/ 29/21	Bit 4/12/ 28/20	Bit 3/11/ 27/19	Bit 2/10/ 26/18	Bit 1/9/ 25/17	Bit 0/8/ 24/16
$1\ 1000_H$ + m x 40_H	FCNnMm DAT0W	FCNnMmSSD[07:00]							
		FCNnMmSSD[17:10]							
		FCNnMmSSD[27:20]							
		FCNnMmSSD[37:30]							
$0\ 9000_H$ + m x 40_H	FCNnMm DAT0H	FCNnMmSSD[07:00]							
		FCNnMmSSD[17:10]							
$0\ 1000_H$ + m x 40_H	FCNnMm DAT0B	FCNnMmSSD[07:00]							
$0\ 1004_H$ + m x 40_H	FCNnMm DAT1B	FCNnMmSSD[17:10]							
$0\ 9008_H$ + m x 40_H	FCNnMm DAT2H	FCNnMmSSD[27:20]							
		FCNnMmSSD[37:30]							
$0\ 1008_H$ + m x 40_H	FCNnMm DAT2B	FCNnMmSSD[27:20]							
$0\ 100C_H$ + m x 40_H	FCNnMm DAT3B	FCNnMmSSD[37:30]							
$1\ 1010_H$ + m x 40_H	FCNnMm DAT4W	FCNnMmSSD[47:40]							
		FCNnMmSSD[57:50]							
		FCNnMmSSD[67:60]							
		FCNnMmSSD[77:70]							
$0\ 9010_H$ + m x 40_H	FCNnMm DAT4H	FCNnMmSSD[47:40]							
		FCNnMmSSD[57:50]							
$0\ 1010_H$ + m x 40_H	FCNnMm DAT4B	FCNnMmSSD[47:40]							
$0\ 1014_H$ + m x 40_H	FCNnMm DAT5B	FCNnMmSSD[57:50]							
$0\ 9018_H$ + m x 40_H	FCNnMm DAT6H	FCNnMmSSD[67:60]							
		FCNnMmSSD[77:70]							
$0\ 1018_H$ + m x 40_H	FCNnMm DAT6B	FCNnMmSSD[67:60]							
$0\ 101C_H$ + m x 40_H	FCNnMm DAT7B	FCNnMmSSD[77:70]							
$0\ 1020_H$ + m x 40_H	FCNnMm DTLGB	0				FCNnMmDTLG[3:0]			
$0\ 1024_H$ + m x 40_H	FCNnMm STRB	FCNnMm SSOW	FCNnMmSSMT[3:0]				FCNnMm SSRT	0	FCNnMm SSAM
$0\ 9028_H$ + m x 40_H	FCNnMm MID0H	FCNnMmSSID[7:0]							
		FCNnMmSSID[15:8]							
$0\ 9030_H$ + m x 40_H	FCNnMm MID1H	FCNnMmSSID[23:16]							
		FCNnMm SSIE	0	0	FCNnMmSSID[28:24]				

<R> **Table 22-17 FCN message buffer register bit configuration (2/2)**

Address offset	Symbol	Bit 7/15/31/23	Bit 6/14/30/22	Bit 5/13/29/21	Bit 4/12/28/20	Bit 3/11/27/19	Bit 2/10/26/18	Bit 1/9/25/17	Bit 0/8/24/16
$1\ 1028_H$ + m x 40_H	FCNnMm MID0W	FCNnMmSSID[7:0]							
		FCNnMmSSID[15:8]							
		FCNnMmSSID[23:16]							
		FCNnMm SSIE	0	0	FCNnMmSSID[28:24]				
$0\ 9038_H$ + m x 40_H	FCNnMmCTL (W)	0	FCNnMm CLNH	0	FCNnMm CLMW	FCNnMm CLIE	FCNnMm CLDN	FCNnMm CLTR	FCNnMm CLRY
		0	FCNnMm SENH	0	0	FCNnMm SEIE	0	FCNnMm SETR	FCNnMm SERY
	FCNnMmCTL (R)	0	FCNnMm NHMF	0	FCNnMm MOWF	FCNnMm IENF	FCNnMm DTNF	FCNnMm TRQF	FCNnMm RDYF
		0	0	FCNnMm MUCF	0	0	0	FCNnMm TCPF	0

22.5 Bit Set/Clear Function

The FCN control registers include registers whose bits can be set or cleared via the CPU and via the CAN Controller. These register bits can not be changed directly by the CPU by any bit manipulation instructions, such as SET1, CLR1, and NOT1. Instead a special bit-set/bit-clear mechanism is used.

All registers where bit manlpulation operations are prohibited are organised in such a way that all bits allowed for changing by the CPU are located in the lower byte (RWx in the register layout below), while in the upper byte either no or read-only information is located (ROx in the register layout below).

The registers can be read in the usual way getting all 16 data bits in their current setting and as described in the register description.
For setting or clearing any of the lower 8 bits the following mechanism is implemented:

When writing 16-bit data to the register address

Bit clear

- each of the lower 8 data bits (CLx in the register layout below) indicates whether the corresponding register bit RWx should be
  - cleared, i.e. set to 0: if CLx = 1, the corresponding RWx is cleared to 0
  - remain unchanged: if CLx = 0, the corresponding RWx does not change

Bit set

- each of the upper 8 data bits (SEx in the register layout below) indicate whether the corresponding register bit should be
  - set, i.e. set to 1: if SEx = 1, the corresponding RWx is set to 1
  - remain unchanged: if SEx = 0, the corresponding RWx does not change

Register layout for read access:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RO7	RO6	RO5	RO4	RO3	RO2	RO1	RO0	RW7	RW6	RW5	RW4	RW3	RW2	RW1	RW0
changing by the CPU not possible								bits for CPU manipulation via SE7-SE0 and CL7-CL0							

Register layout for write access:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE7	SE6	SE5	SE4	SE3	SE2	SE1	SE0	CL7	CL6	CL5	CL4	CL3	CL2	CL1	CL0
SEx = 1 sets the corresponding RW7-RW0								CLx = 1 clears the corresponding RW7-RW0							

The following table denotes the operations applied to the RWx bits:

Table 22-18 Bit set/clear operation

CLx	SEx	Operation on RWx
0	0	Not changed
0	1	Set
1	0	Cleared
1	1	Not changed

Example The following shows an example.

Changing the register with the content 1883_H as follows:

- Bit 3 shall be set: SE3 = 1
- Bit 1 shall be cleared: CL1 = 1

Register read before bit manipulations:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	0	1	0	0	0	**0**	0	**1**	1
may hold any value, here 18_H								RW7-RW0: 83_H							

Register write access:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	**1**	0
SE3 = 1: 08_H								CL1 = 1: 02_H							

Register read after bit manipulations:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	0	1	0	0	0	**1**	0	**0**	1
may hold any value, here 18_H								RW7-RW0: 89_H							

22.6 Control Registers

22.6.1 FCN global registers

(1) FCNnGMCLCTL - FCNn global control register

This register is used to control the operation of the FCN module.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + 0 8000_H

Initial Value $00x0_H$.[a] The register is initialized by any reset.

a) Soft reset starts automatically after hard reset.
So initial value is:
--- If error is not detected after soft reset, then it is 0000_H.
--- If error is not detected on soft reset, then it is 0010_H.
--- If error is detected after soft reset, then it is 0020_H.
--- If error is detected on soft reset, then it is 0030_H.

(a) FCNnGMCLCTL read

15	14	13	12	11	10	9	8
FCNnGMCLSSMO	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	FCNnGMCLECCF	FCNnGMCLSORF	0	0	FCNnGMCLESDE	FCNnGMCLPWOM

FCNnGMCLSSMO	Bit enabling access to message buffer register, transmit/receive history registers
0	Write access and read access to the message buffer register and the transmit/receive history list registers is disabled.
1	Write access and read access to the message buffer register and the transmit/receive history list registers is enabled.

Cautions

1. While the FCNnGMCLCTL.FCNnGMCLSSMO is cleared (to 0), software access to FCN message buffer registers (i.e. all FCNnMm registers) and registers related to transmit history or receive history (FCNnCMLOSTR, FCNnCMTGTX, FCNnCMLISTR, and FCNnCMRGRX) is disabled.
2. FCNnGMCLCTL.FCNnGMCLSSMO is read-only. Even if 1 is written while it is 0, its value does not change, and access to the message buffer registers, or registers related to transmit history or receive history remains disabled.

Note FCNnGMCLCTL.FCNnGMCLSSMO is cleared to (0) when the FCN module enters FCN sleep mode/FCN stop mode, or when the FCNnGMCLCTL.FCNnGMCLPWOM is cleared to (0).
FCNnGMCLSSMO is set to (1) when the FCN sleep mode/FCN stop mode is released, or when the FCNnGMCLCTL.FCNnGMCLPWOM is set to (1).

FCNnGMCLECCF	Message buffer RAM read error detect bit
0	Not detect error for reading from message buffer RAM.
1	Detect error for reading from message buffer RAM.

Notes

1. FCNnGMCLCTL.FCNnGMCLECCF is set (1) in case of detecting a memory error when reading from the message buffer RAM during the soft reset process. Once FCNnGMCLECCF is set (1), it keeps the level until it is cleared (0).
2. Only use this bit to check for memory errors after executing a soft reset.
3. It is impossible to clear FCNnGMCLECCF (0) during FCNnGMCLCTL.FCNnGMCLSORF is set (1) (soft reset is ongoing).

FCNnGMCLSORF	Soft reset execution status bit
0	No soft reset
1	Soft reset is ongoing

Notes

1. While a soft reset is ongoing (FCNnGMCLCTL.FCNnGMCLSORF is set (1)), it is impossible to set FCNnGMCLCTL.FCNnGMCLPWOM and FCNnGMCLCTL.EFSD.
   It is possible to set start a software reset by FCNnGMCLCTL.FCNnGMCLSESR = 1 during FCNnGMCLCTL.FCNnGMCLPWOM bit is clear (0).
2. When FCNnGMCLCTL.FCNnGMCLSORF is set (1), the initialization of message buffer RAM starts. It is possible to detect error during initializing message buffer RAM, if FCNnGMCLCTL.FCNnGMCLECCF is cleared before setting FCNnGMCLSORF.
3. When FCNnGMCLCTL.FCNnGMCLSORF is set (1) again in the condition that is is already set (1), the soft reset procedure does not restart, but continues.
4. After releasing hardware reset FCNnGMCLCTL.FCNnGMCLSORF is set (1) automatically and initialization of message buffer RAM starts.
5. It is impossible that clearing FCNnGMCLCTL.FCNnGMCLPWOM (0) and setting FCNnGMCLCTL.FCNnGMCLSORF (1) are done at the same time.
6. If a hardware RESET occurs during FCNnGMCLCTL.FCNnGMCLSORF = 1, then the soft reset procedure is stopped (aborted), and the hardware RESET starts.

FCNnGMCLESDE	Bit enabling forced shut down
0	Forced shutdown by setting FCNnGMCLCTL.FCNnGMCLPWOM = 0 is disabled.
1	Forced shutdown by setting FCNnGMCLCTL.FCNnGMCLPWOM = 0 is enabled.

Caution To request a forced shut down, FCNnGMCLCTL.FCNnGMCLPWOM must be cleared to 0 in a subsequent, immediately following access after FCNnGMCLCTL.FCNnGMCLESDE has been set to 1. If any access to another register (including reading the FCNnGMCLCTL register) is executed without clearing FCNnGMCLPWOM immediately after FCNnGMCLESDE has been set to 1, FCNnGMCLESDE is forcibly cleared to 0, and the forced shut down request is invalid.

FCNnGMCLPWOM	Global operation mode bit
0	FCN module is disabled.
1	FCN module is enabled to operate.

Caution FCNnGMCLCTL.FCNnGMCLPWOM can be cleared only in the initialization mode or immediately after FCNnGMCLCTL.FCNnGMCLESDE is set (forced shutdown).

(b) FCNnGMCLCTL write

15	14	13	12	11	10	9	8
0	0	0	FCNnGMCLSESR	0	0	FCNnGMCLSESD	FCNnGMCLSEOM
7	6	5	4	3	2	1	0
0	0	FCNnGMCLCLMB	0	0	0	0	FCNnGMCLCLOM

FCNnGMCLSESR	Software reset start
0	No changes.
1	Start soft reset.

FCNnGMCLSESD	FCNnGMCLESDE bit setting
0	No change in FCNnGMCLESDE bit.
1	FCNnGMCLESDE bit set to 1.

FCNnGMCLSEOM	FCNnGMCLCLOM	FCNnGMCLPWOM bit setting
0	1	FCNnGMCLCTL.FCNnGMCLPWOM bit cleared to 0.
1	0	FCNnGMCLCTL.FCNnGMCLPWOM bit set to 1.
Other than above		No change of FCNnGMCLCTL.FCNnGMCLPWOM bit.

Caution Set FCNnGMCLCTL.FCNnGMCLPWOM and FCNnGMCLCTL.FCNnGMCLESDE bit always separately.

FCNnGMCLCLMB	FCNnGMCLCTL.FCNnGMCLECCF bit clear
0	No change in FCNnGMCLCTL.FCNnGMCLECCF bit.
1	FCNnGMCLCTL.FCNnGMCLECCF bit cleared to 0.

(2) FCNnGMCSPRE - FCNn global clock selection register

This register is used to select the FCN module system clock.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + 0008_H

Initial Value $0F_H$. The register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	FCNnGMCSPRSC[3:0]			

<R>

FCNnGMCSPRSC[3:0]	Pre-CAN protocol layer basic system clock (f_{CANPRE})
0000_B	$f_{CAN}/1$
0001_B	$f_{CAN}/2$
0010_B	$f_{CAN}/3$
0011_B	$f_{CAN}/4$
0100_B	$f_{CAN}/5$
0101_B	$f_{CAN}/6$
0110_B	$f_{CAN}/7$
0111_B	$f_{CAN}/8$
1000_B	$f_{CAN}/9$
1001_B	$f_{CAN}/10$
1010_B	$f_{CAN}/11$
1011_B	$f_{CAN}/12$
1100_B	$f_{CAN}/13$
1101_B	$f_{CAN}/14$
1110_B	$f_{CAN}/15$
1111_B	$f_{CAN}/16$ (default value)

Note f_{CAN} = clock supplied to FCN on system level (clock generation, distribution and selection).

(3) FCNnGMABCTL - FCNn global automatic block transmission control register

This register is used to control the automatic block transmission (ABT) operation.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8018_H$

Initial Value 0000_H. The register is initialized by any reset.

(a) FCNnGMABCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	FCNnGMABCLRF	FCNnGMABABTT

FCNnGMABCLRF	Automatic block transmission engine clear status bit
0	Clearing the automatic transmission engine is completed.
1	The automatic transmission engine is being cleared.

<R> **Note** Be sure to set FCNnGMABCLR to 1 while FCNnGMABABTT is 0. The operation is not guaranteed if FCNnGMABCLRF is set to 1 while FCNnGMABABTT is 1.

FCNnGMABABTT	Automatic block transmission status bit
0	Automatic block transmission is stopped.
1	Automatic block transmission is under execution.

(b) FCNnGMABCTL write

15	14	13	12	11	10	9	8
0	0	0	0	0	0	FCNnGMABSEAC	FCNnGMABSEAT
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FCNnGMABCLAT

Note When the automatic block transmission engine is cleared by setting FCNnGMABCTL.FCNnGMABSEAC to 1, FCNnGMABCLRF is automatically set, and cleared to 0 as soon as the requested clearing processing is completed.

Cautions

1. Before changing the normal operation mode with ABT to the initialization mode, be sure to set the FCNnGMABCTL register to the default value (0000_H) and confirm the FCNnGMABCTL register is surely initialized to the default value (0000_H).
2. Do not start automatic block transmission in the initialization mode. If automatic block transmission is started in the initialization mode, the operation is not guaranteed after the CAN Controller has entered the normal operation mode with ABT.
3. Do not start automatic block transmission while FCNnCMCLCTL.FCNnCMCLSSTS is set to 1 (transmission in progress). Confirm FCNnCMCLSSTS = 0 directly in advance before starting automatic block transmission.

FCNnGMABSEAC	Automatic block transmission engine clear request bit
0	The automatic block transmission engine is in idle status or under operation.
1	Request to clear the automatic block transmission engine. After the automatic block transmission engine has been cleared, automatic block transmission is started from message buffer 0 by setting the FCNnGMABCTL.FCNnGMABABTT = 1.

FCNnGMABSEAT	FCNnGMABCLAT	Automatic block transmission start bit
0	1	Request to stop automatic block transmission.
1	0	Request to start automatic block transmission.
Other than above		No change of FCNnGMABCTL.FCNnGMABABTT.

(4) FCNnGMADCTL - FCNn global automatic block transmission delay register

This register is used to set the interval at which the data of the message buffer assigned to ABT is to be transmitted in the normal operation mode with ABT.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + 0020_H

Initial Value 00_H. The register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	FCNnGMADSSAD[3:0]			

FCNnGMADSSAD[3:0]	Data frame interval during automatic block transmission in DBT[a]
0000_B	0 DBT (default value)
0001_B	2^5 DBT
0010_B	2^6 DBT
0011_B	2^7 DBT
0100_B	2^8 DBT
0101_B	2^9 DBT
0110_B	2^{10} DBT
0111_B	2^{11} DBT
1000_B	2^{12} DBT
Other than above	Setting prohibited

a) Unit: Data bit time (DBT)

Cautions

1. Do not change the contents of the FCNnGMADCTL register while FCNnGMABCTL.FCNnGMABCLRF = 1 (clearing of ABT in progress).
2. The timing at which the ABT message is actually transmitted onto the CAN bus differs depending on the status of transmission from the other station or how a request to transmit a message other than an ABT message is made.

(5) FCNnDNBMRXk – FCNn global data update bit monitor register (k = 0 to 3)

These registers are used to read the data update bits of several message buffers at a time, globally.

Access These registers can be read in 32-bit units.

Address FCNnDNBMRX0: <FCNn_base> + 1 $00C0_H$
FCNnDNBMRX1: <FCNn_base> + 1 $00D0_H$
Following registers are available only with m = 128 message buffers:
FCNnDNBMRX2: <FCNn_base> + 1 $00E0_H$
FCNnDNBMRX3: <FCNn_base> + 1 $00F0_H$

Initial Value 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24
FCNnDNBMSSDN[31:24]							

23	22	21	20	19	18	17	16
FCNnDNBMSSDN[23:16]							

15	14	13	12	11	10	9	8
FCNnDNBMSSDN[15:8]							

7	6	5	4	3	2	1	0
FCNnDNBMSSDN[7:0]							

FCNnDNBMSSDN[31:0]	Message buffer data update bit
0	No remote or data frame has been stored into the message buffer.
1	A remote or data frame has been stored into the message buffer.

22.6.2 FCN module registers

(1) FCNnCMMKCTLaH - FCNn module mask control register

These registers are used to extend the number of receivable messages into the same message buffer by masking part of the identifier (ID) comparison of a message and invalidating the ID of the masked part.

Two 16-bit registers FCNnCMMKCTLaH (a = 01 to 16) can also be accessed via a single 32-bit access to the registers FCNnCMMKCTLaW (a = 01, 03, 05, 07, 09, 11, 13, 15).

Access The FCNnCMMKCTLaH registers can be read/written in 16-bit units.
The FCNnCMMKCTLaW registers can be read/written in 32-bit units.

Address

FCNnCMMKCTL01H: <FCNn_base> + $0\ 8300_H$
FCNnCMMKCTL02H: <FCNn_base> + $0\ 8308_H$
FCNnCMMKCTL03H: <FCNn_base> + $0\ 8310_H$
FCNnCMMKCTL04H: <FCNn_base> + $0\ 8318_H$
FCNnCMMKCTL05H: <FCNn_base> + $0\ 8320_H$
FCNnCMMKCTL06H: <FCNn_base> + $0\ 8328_H$
FCNnCMMKCTL07H: <FCNn_base> + $0\ 8330_H$
FCNnCMMKCTL08H: <FCNn_base> + $0\ 8338_H$
FCNnCMMKCTL09H: <FCNn_base> + $0\ 8340_H$
FCNnCMMKCTL10H: <FCNn_base> + $0\ 8348_H$
FCNnCMMKCTL11H: <FCNn_base> + $0\ 8350_H$
FCNnCMMKCTL12H: <FCNn_base> + $0\ 8358_H$
FCNnCMMKCTL13H: <FCNn_base> + $0\ 8360_H$
FCNnCMMKCTL14H: <FCNn_base> + $0\ 8368_H$
FCNnCMMKCTL15H: <FCNn_base> + $0\ 8370_H$
FCNnCMMKCTL16H: <FCNn_base> + $0\ 8378_H$

FCNnCMMKCTL01W: <FCNn_base> + $1\ 0300_H$
FCNnCMMKCTL03W: <FCNn_base> + $1\ 0310_H$
FCNnCMMKCTL05W: <FCNn_base> + $1\ 0320_H$
FCNnCMMKCTL07W: <FCNn_base> + $1\ 0330_H$
FCNnCMMKCTL09W: <FCNn_base> + $1\ 0340_H$
FCNnCMMKCTL11W: <FCNn_base> + $1\ 0350_H$
FCNnCMMKCTL13W: <FCNn_base> + $1\ 0360_H$
FCNnCMMKCTL15W: <FCNn_base> + $1\ 0370_H$

<R> **Initial Value** 0000_H for FCNnCMMKCTLaH. This register is initialized by any reset.
$0000\ 0000_H$ for FCNnCMMKCTLaW. This register is initialized by any reset.

(a) FCNnCMMKCTLaH (a = 01, 03, 05, 07, 09, 11, 13, 15)

15	14	13	12	11	10	9	8
FCNnCMMKSSID[15:8]							

7	6	5	4	3	2	1	0
FCNnCMMKSSID[7:0]							

(b) FCNnCMMKCTLaH (a = 02, 04, 06, 08, 10, 12, 14, 16)

15	14	13	12	11	10	9	8
0	0	0	FCNnCMMKSSID[28:24]				

7	6	5	4	3	2	1	0
FCNnCMMKSSID[23:16]							

(c) FCNnCMMKCTLaW (a = 01, 03, 05, 07, 09, 11, 13, 15)

31	30	29	28	27	26	25	24
			FCNnCMMKSSID[28:24]				

23	22	21	20	19	18	17	16
FCNnCMMKSSID[23:16]							

15	14	13	12	11	10	9	8
FCNnCMMKSSID[15:8]							

7	6	5	4	3	2	1	0
FCNnCMMKSSID[7:0]							

FCNnCMMKSSID[i] [a]	Mask pattern setting of ID bit
0	The ID bit **i** of the message buffer m set by FCNnMmSSID[**i**] are compared with the ID bits of the received message frame.
1	The ID bit **i** of the message buffer m set by FCNnMmSSID[**i**] are not compared with the ID bits of the received message frame (they are masked).

a) i = [28:0]

Note Masking is always defined by an ID length of 29 bits. If a mask is assigned to a message with a standard ID, FCNnCMMKSSID[17:0] are ignored. Therefore, only FCNnCMMKSSID[28:18] of the received ID are masked. The same mask can be used for both the standard and extended IDs.

(2) FCNnCMCLCTL - FCNn module control register

This register is used to control the operation mode of the FCN module.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8240_H$

Initial Value 0000_H. The register is initialized by any reset.

(a) FCNnCMCLCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	FCNnCMCLSSRS	FCNnCMCLSSTS

7	6	5	4	3	2	1	0
FCNnCMCLERCF	FCNnCMCLALBF	FCNnCMCLVALF	FCNnCMCLMDPF[1:0]		FCNnCMCLMDOF[2:0]		

FCNnCMCLSSRS	Reception status bit
0	Reception is stopped.
1	Reception is in progress.

Notes

1. FCNnCMCLSSRS is set to 1 under the following conditions (timing)
   - The SOF bit of a receive frame is detected
   - On occurrence of arbitration loss during a transmit frame
2. FCNnCMCLSSRS is cleared to 0 under the following conditions (timing)
   - When a recessive level is detected at the second bit of the interframe space
   - On transition to the initialization mode at the first bit of the interframe space

FCNnCMCLSSTS	Transmission status bit
0	Transmission is stopped.
1	Transmission is in progress.

Notes

1. FCNnCMCLSSTS is set to 1 under the following conditions (timing)
   - The SOF bit of a transmit frame is detected
2. FCNnCMCLSSTS is cleared to 0 under the following conditions (timing)
   - During transition to bus-off state
   - On occurrence of arbitration loss in transmit frame
   - On detection of recessive level at the second bit of the interframe space
   - On transition to the initialization mode at the first bit of the interframe space

FCNnCMCLERCF	Error counter clear bit
<R> 0	The FCNnCMERCNT and FCNnCMINSTR registers are not cleared in the initialization mode.
<R> 1	The FCNnCMERCNT and FCNnCMINSTR registers are cleared in the initialization mode.

<R> **Caution** FCNnCMCLERCF is used to clear the error counter FCNnCMERCNT and information register FCNnCMINSTR for re-initialization or forced recovery from the bus-off state. The error counter and the information register can be cleared under the following conditions (by setting FCNnCMCLERCF):

- In the initialization mode during the bus-off period
- In the initialization mode after the FCN module starts up (by changing FCNnGMCLPWOM from 0 to 1)
- In the initialization mode entered after all the transmission requests have been cleared in accordance with the transmission abort processing shown in *Table 22-25 "Transmission abort processing (except normal operation mode with ABT)"* in an operation mode. (In normal operation mode with ABT, clear all the transmission requests in accordance with the transmission abort processing shown in *Table 22-26 "Processing to abort transmission other than ABT transmission (in normal operation mode with ABT)"*.)

<R> **Notes**

1. When the FCNnCMERCNT and FCNnCMINSTR registers have been cleared, FCNnCMCLERCF is also cleared to 0 automatically.
2. FCNnCMCLERCF can be set to 1 at the same time as a request to change the initialization mode to an operation mode is made.
3. FCNnCMCLERCF is read-only in the FCN sleep mode or FCN stop mode.
4. <R> The error counter can also be cleared by a normal shutdown or forced shutdown of the CAN controller.

FCNnCMCLALBF	Bit to set operation in case of arbitration loss
0	Re-transmission is not executed in case of an arbitration loss in the single-shot mode.
1	Re-transmission is executed in case of an arbitration loss in the single-shot mode.

Note FCNnCMCLALBF is valid only in the single-shot mode.

FCNnCMCLVALF	Valid receive message frame detection bit
0	A valid message frame has not been received since FCNnCMCLVALF was last cleared to 0.
1	A valid message frame has been received since FCNnCMCLVALF was last cleared to 0.

Notes

1. Detection of a valid receive message frame is not dependent upon storage in the receive message buffer (data frame/remote frame) or transmit message buffer (remote frame).
2. If only two CAN nodes are connected to the CAN bus with one transmitting a message frame in the normal mode and the other in the receive-only mode, FCNnCMCLVALF is not set to 1 before the transmitting node enters the error passive state, because in receive-only mode no acknowledge is generated.
3. To clear FCNnCMCLVALF, set FCNnCMCLCLVL to 1 first and confirm that FCNnCMCLVALF is cleared. If it is not cleared, perform clearing processing again.

FCNnCMCLMDPF[1:0]	Power save mode
00_B	No power save mode is selected.
01_B	FCN sleep mode
10_B	Setting prohibited
11_B	FCN stop mode

Cautions

1. Transition to and from the FCN stop mode must be made via FCN sleep mode. A request for direct transition to and from the FCN stop mode is ignored.
2. The FCNnGMCLSSMO flag of FCNnGMCLCTL must be checked after releasing a power save mode, prior to access the message buffers again.
3. FCN sleep mode requests are kept pending, until cancelled by software or entered on appropriate bus condition (bus idle). Software can check the actual status by reading FCNnCMCLMDPF[1:0].

<R>

4. Power save mode cannot be set in combination with the change of operation mode. Be sure to perform these operations in different steps.

Note When the system transitions from initialization mode to a communication mode, the FCN module participates in communication after first confirming the CAN bus idle period. Although it is possible to transition to sleep mode before the idle period has been confirmed, in this case, the wakeup condition will always change from recessive level to dominant level.

FCNnCMCLMDOF[2:0]	Operation mode
000_B	No operation mode is selected (FCN module is in the initialization mode).
001_B	Normal operation mode
010_B	Normal operation mode with automatic block transmission function (normal operation mode with ABT)
011_B	Receive-only mode
100_B	Single-shot mode
101_B	Self-test mode
Other than above	Setting prohibited

Cautions

1. Transition to initialization mode or power saving modes may take some time. Be sure to verify the success of mode change by reading the values, before proceeding.

<R>

2. If initialization mode is set while receiving data in operation mode, data in the message buffer that sets the FCNnMmDTNF flag might be received last. However, the receive history list is cleared upon transition to operation mode. It is therefore necessary to confirm that initialization mode was set by reading the operation mode. Before restarting operation mode, make sure to clear all FCNnMmDTNF flags in every valid reception message buffer.

Note FCNnCM.FCNnCMCLMDOF[2:0] are read-only in the FCN sleep mode or FCN stop mode.

(b) FCNnCMCLCTL write

15	14	13	12	11	10	9	8
FCNnCM CLSERC	FCNnCM CLSEAL	0	FCNnCM CLSEPS[1:0]		FCNnCM CLSEOP[2:0]		
7	6	5	4	3	2	1	0
0	FCNnCM CLCLAL	FCNnCM CLCLVL	FCNnCM CLCLPS[1:0]		FCNnCM CLCLOP[2:0]		

FCNnCMCLSERC	Setting of FCNnCMCLERCF bit
1	FCNnCMCLERCF is set to 1.
Other than above	FCNnCMCLERCF is not changed.

FCNnCMCLSEAL	FCNnCMCLCLAL	Setting of FCNnCMCLALBF bit
0	1	FCNnCMCLALBF is cleared to 0.
1	0	FCNnCMCLALBF is set to 1.
Other than above		FCNnCMCLALBF is not changed.

FCNnCMCLCLVL	Setting of FCNnCMCLVALF bit
0	FCNnCMCLVALF is not changed.
1	FCNnCMCLVALF is cleared to 0.

FCNnCMSESEPS0	FCNnCMCLCLPS0	Setting of FCNnCMCLMDPF0 bit
0	1	FCNnCMCLMDPF0 is cleared to 0.
1	0	FCNnCMCLMDPF0 is set to 1.
Other than above		FCNnCMCLMDPF0 is not changed.

FCNnCMSESEPS1	FCNnCMCLCLPS1	Setting of FCNnCMCLMDPF1 bit
0	1	FCNnCMCLMDPF1 is cleared to 0.
1	0	FCNnCMCLMDPF1 is set to 1.
Other than above		FCNnCMCLMDPF1 is not changed.

FCNnCMCLSEOP0	FCNnCMCLCLOP0	Setting of FCNnCMCLMDOF0 bit
0	1	FCNnCMCLMDOF0 is cleared to 0.
1	0	FCNnCMCLMDOF0 is set to 1.
Other than above		FCNnCMCLMDOF0 is not changed.

FCNnCMCLSEOP1	FCNnCMCLCLOP1	Setting of FCNnCMCLMDOF1 bit
0	1	FCNnCMCLMDOF1 is cleared to 0.
1	0	FCNnCMCLMDOF1 is set to 1.
Other than above		FCNnCMCLMDOF1 is not changed.

FCNnCMCLSEOP2	FCNnCMCLCLOP2	Setting of FCNnCMCLMDOF2 bit
0	1	FCNnCMCLMDOF2 is cleared to 0.
1	0	FCNnCMCLMDOF2 is set to 1.
Other than above		FCNnCMCLMDOF2 is not changed.

(3) FCNnCMLCSTR - FCNn module last error information register

This register provides the error information of the CAN protocol.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + 0248_H

Initial Value 00_H. The register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	FCNnCMLCSSLC[2:0]		

Notes

1. The contents of the FCNnCMLCSTR register are not cleared when the FCN module changes from an operation mode to the initialization mode.
2. If an attempt is made to write a value other than 00_H to the FCNnCMLCSTR register by software, the access is ignored.

FCNnCMLCSSLC[2:0]	Last CAN protocol error information
000_B	No error
001_B	Stuff error
010_B	Form error
011_B	ACK error
100_B	Bit error. (The FCN module tried to transmit a recessive-level bit as part of a transmit message (except the arbitration field), but the value on the CAN bus is a dominant-level bit.)
101_B	Bit error. (The FCN module tried to transmit a dominant-level bit as part of a transmit message, ACK bit, error frame, or overload frame, but the value on the CAN bus is a recessive-level bit.)
110_B	CRC error
111_B	Undefined

(4) FCNnCMINSTR - FCNn module information register

This register indicates the status of the FCN module.

Access This register is read-only in 8-bit units.

Address <FCNn_base> + 0 $024C_H$

Initial Value 00_H. The register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	FCNnCM INBOFF	FCNnCM INSSTE[1:0]		FCNnCM INSSRE[1:0]	

FCNnCMINBOFF	Bus-off state bit
0	Not bus-off state (transmit error counter ≤ 255). (The value of the transmit counter is less than 256.)
1	Bus-off state (transmit error counter > 255). (The value of the transmit counter is 256 or more.)

FCNnCMINSSTE[1:0]	Transmission error counter status bit
00_B	The value of the transmission error counter is less than that of the warning level (< 96).
01_B	The value of the transmission error counter is in the range of the warning level (96 to 127).
10_B	Undefined
11_B	The value of the transmission error counter is in the range of the error passive or bus-off status (≥ 128).

FCNnCMINSSRE[1:0]	Reception error counter status bit
00_B	The value of the reception error counter is less than that of the warning level (< 96).
01_B	The value of the reception error counter is in the range of the warning level (96 to 127).
10_B	Undefined
11_B	The value of the reception error counter is in the error passive range (≥ 128).

(5) FCNnCMERCNT - FCNn module error counter register

This register indicates the count value of the transmission/reception error counter.

Access This register is read-only in 16-bit units.

Address <FCNn_base> + $0\ 8250_H$

Initial Value 0000_H. The register is initialized by any reset.

15	14	13	12	11	10	9	8
FCNnCM ERRPSF	FCNnCM ERRECF[6:0]						
7	6	5	4	3	2	1	0
FCNnCM ERTECF[7:0]							

FCNnCMERRPSF	Reception error passive status bit
0	The reception error counter is not in the error passive range (< 128)
1	The reception error counter is in the error passive range (≥ 128)

FCNnCMERRECF[6:0]	Reception error counter bit
0 to 127	The reception error count. These bits reflect the status of the reception error counter. The error count is defined by the CAN protocol.

Note FCNnCMERRECF[6:0] are invalid in the reception error passive state (FCNnCMINSTR.FCNnCMINSSRE[1:0] = 11_B).

FCNnCMERTECF[7:0]	Transmission error counter bit
0 to 255	Number of transmission errors. These bits reflect the status of the transmission error counter. The number of errors is defined by the CAN protocol.

Note FCNnCMERTECF[7:0] are invalid in the bus-off state (FCNnCMINSTR.FCNnCMINBOFF = 1).

(6) FCNnCMIECTL - FCNn module interrupt enable register

This register is used to enable or disable the interrupts of the FCN module.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8258_H$

Initial Value 0000_H. The register is initialized by any reset.

(a) FCNnCMIECTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	FCNnCMIEINTF[6:0]						

FCNnCMIEINTF[6:0]	FCN module interrupt enable bit
0	Output of the interrupt corresponding to interrupt status register FCNnCMISCTL is disabled.
1	Output of the interrupt corresponding to interrupt status register FCNnCMISCTL is enabled.

(b) FCNnCMIECTL write

15	14	13	12	11	10	9	8
0	FCNnCMIESEIE[6:0]						
7	6	5	4	3	2	1	0
0	FCNnCMIECLIE[6:0]						

FCNnCMIESEIE[6:0]	FCNnCMIECLIE[6:0]	Setting of FCNnCMIEINTF[6:0] bit
0	1	FCNnCMIEINTF[6:0] bit is cleared to 0.
1	0	FCNnCMIEINTF[6:0] bit is set to 1.
Other than above		FCNnCMIEINTF[6:0] bit is not changed.

(7) FCNnCMISCTL - FCNn module interrupt status register

This register indicates the interrupt status of the FCN module.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8260_H$

Initial Value 0000_H. The register is initialized by any reset.

(a) FCNnCMISCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	FCNnCMISITSF[6:0]						

FCNnCMISITSF[6:0]	FCN interrupt status bit
0	No related interrupt source event is pending
1	A related interrupt source event is pending

Interrupt status bit	Related interrupt source event
FCNnCMISITSF6	FCN module transmission abort interrupt status bit
FCNnCMISITSF5	Wakeup interrupt from FCN sleep mode [a]
FCNnCMISITSF4	Arbitration loss interrupt
FCNnCMISITSF3	CAN protocol error interrupt
FCNnCMISITSF2	CAN error status interrupt
FCNnCMISITSF1	Interrupt on completion of reception of valid message frame to message buffer m
FCNnCMISITSF0	Interrupt on normal completion of transmission of message frame from message buffer m

a) FCNnCMISITSF5 is set only when the FCN module is woken up from the FCN sleep mode by a CAN bus operation. It is not set when the FCN sleep mode has been released by software.

(b) FCNnCMISCTL write

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	FCNnCMISITSF[6:0]						

<R>

FCNnCMISCLTS[6:0]	Clearing of FCNnCMISITSF[6:0]
0	FCNnCMISITSF[6:0] bits are not changed.
1	FCNnCMISITSF[6:0] bits are cleared to 0.

Caution Clear the status bit of this register by software, when the confirmation of each status is necessary in the interrupt processing, because these bits are not cleared automatically.

(8) FCNnCMBRPRS - FCNn module bit rate prescaler register

This register is used to select the CAN protocol layer basic system clock (f_{TQ}). The communication baud rate is set to the FCNnCMBTCTL register.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + $0\ 0268_H$

Initial Value FF_H. The register is initialized by any reset.

7	6	5	4	3	2	1	0
FCNnCMBRPRS[7:0]							

FCNnCMBRPRS[7:0]	CAN protocol layer basic system clock (f_{TQ})
0	$f_{CANPRE}/1$
1	$f_{CANPRE}/2$
n	$f_{CANPRE}/(n+1)$
:	:
255	$f_{CANPRE}/256$ (default value)

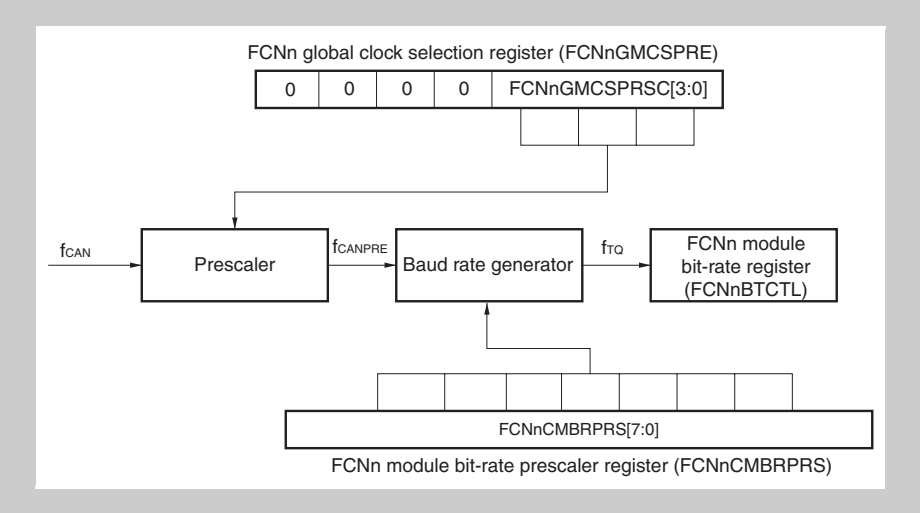


Figure 22-3 FCN module clock

Note f_{CAN}: Clock supplied to FCN
f_{CANPRE}: Pre-CAN protocol layer basic system clock
f_{TQ}: CAN protocol layer basic system clock

Caution FCNnCMBRPRS can be write-accessed only in the initialization mode.

(9) FCNnCMBTCTL - FCNn module bit rate register

This register is used to control the data bit time of the communication baud rate.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8270_H$

Initial Value $370F_H$. The register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	FCNnCM BTJWLG[1:0]		0	FCNnCM BTS2LG[2:0]		
7	6	5	4	3	2	1	0
0	0	0	0	FCNnCMBTS1LG[3:0]			

<R>

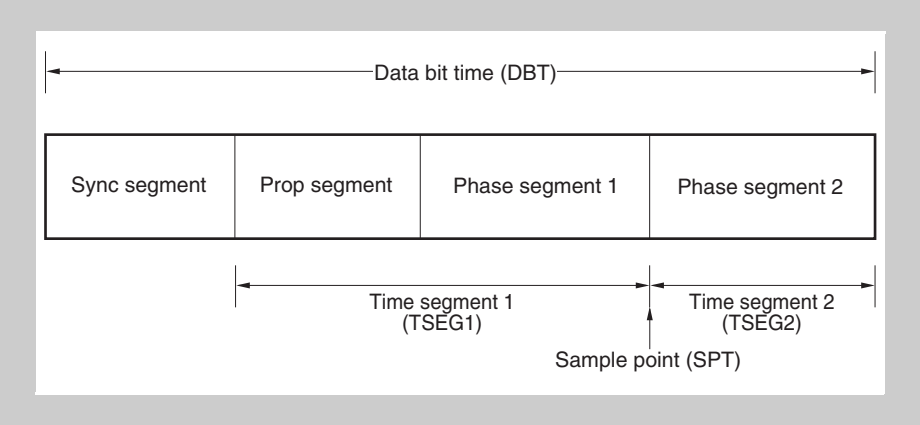


Figure 22-4 Data bit time

<R>

FCNnCMBTJWLG[1:0]	Length of synchronization jump width (SJW)
00_B	$1T_Q$
01_B	$2T_Q$
10_B	$3T_Q$
11_B	$4T_Q$ (default value)

<R>

FCNnCMBTS2LG[2:0]	Length of time segment 2 (TSEG2)
000_B	$1T_Q$
001_B	$2T_Q$
010_B	$3T_Q$
011_B	$4T_Q$
100_B	$5T_Q$
101_B	$6T_Q$
110_B	$7T_Q$
111_B	$8T_Q$ (default value)

<R>

FCNnCMBTS1LG[3:0]	Length of time segment 1 (TSEG1)
0000_B	Setting prohibited
0001_B	$2T_Q$[a]
0010_B	$3T_Q$[a]
0011_B	$4T_Q$
0100_B	$5T_Q$
0101_B	$6T_Q$
0110_B	$7T_Q$
0111_B	$8T_Q$
1000_B	$9T_Q$
1001_B	$10T_Q$
1010_B	$11T_Q$
1011_B	$12T_Q$
1100_B	$13T_Q$
1101_B	$14T_Q$
1110_B	$15T_Q$
1111_B	$16T_Q$ (default value)

a) This setting must not be made when the FCNnCMBRPRS register = 00_H

Note $T_Q = 1/f_{TQ}$ (f_{TQ}: CAN protocol layer basic system clock)

(10) FCNnCMLISTR - FCNn module last in-pointer register

This register indicates the number of the message buffer in which a data frame or a remote frame was last stored.

Access This register is read-only in 8-bit units.

Address <FCNn_base> + 0 0278_H

Initial Value Undefined.

7	6	5	4	3	2	1	0
FCNnCMLISSLR[7:0]							

FCNnCMLISSLR[7:0]	Last in-pointer register of receive history list
0 to 63 [a] 0 to 127 [b]	Reading the FCNnCMLISTR register obtains the number of the message buffer storing the last data frame or remote frame to be received.

a) On 64 message buffer FCN.
b) On 128 message buffer FCN.

Note The read value of FCNnCMLISTR is undefined if a data frame or a remote frame has never been received and stored in the message buffer. If FCNnCMRGRX.FCNnCMRGSSPM is set to 1 after the FCN module has changed from the initialization mode to an operation mode, therefore, the read value of FCNnCMLISTR is undefined.

(11) FCNnCMRGRX - FCNn module receive history list register

This register is used to read the receive history list (RHL).

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + 0 8280_H

Initial Value $xx02_H$. The register is initialized by any reset.

(a) FCNnCMRGRX read

15	14	13	12	11	10	9	8
FCNnCMRGSSPT[7:0]							
7	6	5	4	3	2	1	0
0	0	0	0	0	0	FCNnCM RGSSPM	FCNnCM RGRVFF

FCNnCMRGSSPT[7:0]	Receive history list read pointer
0 to 63 [a] 0 to 127 [b]	When FCNnCMRGRX is read, the contents of the element indexed by the receive history list get pointer (FCNnCMRGRX.FCNnCMRGSSPT) of the receive history list are read. These contents indicate the number of the message buffer in which a data frame or a remote frame has been stored.

a) On 64 message buffer FCN.
b) On 128 message buffer FCN.

FCNnCMRGSSPM[a]	Receive history list pointer match
0	The receive history list has at least one message buffer number that has not been read.
1	The receive history list has no message buffer numbers that have not been read.

a) The read value of FCNnCMTGSSPT[7:0] is invalid when FCNnCMTGSSPM = 1.

FCNnCMRGRVFF[a]	Receive history list overflow bit [b]
0	All the message buffer numbers that have not been read are preserved. All the numbers of the message buffers in which a new data frame or remote frame has been received and stored are recorded to the receive history list (the receive history list has a vacant element).
1	At least **(i)** entries have been stored since the host processor has serviced the RHL last time (i.e. read FCNnCMRGRX). The first **(i-1)** entries are sequentially stored while the last entry can have been overwritten whenever newly received message is stored, because all buffer numbers are stored at position **(i)**, when FCNnCMRGRVFF is set. Thus the sequence of receptions can not be recovered completely now.

a) If all the receive history entries have been read by the FCNnCMRGRX register while FCNnCMRGRVFF is set (1), FCNnCMRGSSPM is not cleared even if a new message is received.

b) i = 47 on 64 message buffer FCN;
i = 95 on 128 message buffer FCN.

(b) FCNnCMRGRX write

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FCNnCMRGCLRV

FCNnCMRGCLRV	Clearing of FCNnCMRGRVFF bit
0	FCNnCMRGRVFF bit is not changed.
1	FCNnCMRGRVFF bit is cleared to 0.

(12) FCNnCMLOSTR - FCNn module last out-pointer register

This register indicates the number of the message buffer, from which a data frame or a remote frame was transmitted last.

Access This register is read-only in 8-bit units.

Address <FCNn_base> + 0288_H

Initial Value Undefined

7	6	5	4	3	2	1	0
FCNnCMLOSSLT[7:0]							

FCNnCMLOSSLT[7:0]	Last out-pointer of transmit history list
0 to 63 [a] 0 to 127 [b]	When the FCNnCMLOSTR register is read, the contents of the element indexed by the last out-pointer (FCNnCMLOSTR[7:0]) of the receive history list are read. These contents indicate the number of the message buffer to which a data frame or a remote frame was transmitted last.

a) On 64 message buffer FCN.
b) On 128 message buffer FCN.

Note The value read from the FCNnCMLOSTR register is undefined if a data frame or remote frame has never been transmitted from a message buffer.

(13) FCNnCMTGTX - FCNn module transmit history list register

This register is used to read the transmit history list (THL).

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + 0 8290_H

Initial Value $xx02_H$. The register is initialized by any reset.

(a) FCNnCMTGTX read

15	14	13	12	11	10	9	8
FCNnCMTGSSPT[7:0]							
7	6	5	4	3	2	1	0
0	0	0	0	0	0	FCNnCMTGSSPM	FCNnCMTGTVFF

FCNnCMTGSSPT[7:0]	Transmit history list read pointer
0 to 63 [a] 0 to 127 [b]	When the FCNnCMTGTX register is read, the contents of the element indexed by the read pointer (FCNnCMTGSSPT[7:0]) of the transmit history list are read. These contents indicate the number of the message buffer to which a data frame or a remote frame was transmitted last.

a) On 64 message buffer FCN.
b) On 128 message buffer FCN.

FCNnCMTGSSPM[a]	Transmit history pointer match
0	The transmit history list has at least one message buffer number that has not been read.
1	The transmit history list has no message buffer numbers that have not been read.

a) The read value of FCNnCMTGSSPT[7:0] is invalid when the FCNnCMTGSSPM = 1.

FCNnCMTGTVFF[a]	Transmit history list overflow bit [b]
0	All the message buffer numbers that have not been read are preserved. All the numbers of the message buffers to which a new data frame or remote frame has been transmitted are recorded to the transmit history list (the transmit history list has a vacant element).
1	At least **(i)** entries have been stored since the host processor has serviced the THL last time (i.e. read FCNnCMTGTX). The first **(i-1)** entries are sequentially stored while the last entry can have been overwritten whenever newly received message is stored, because all buffer numbers are stored at position **(i)**, when FCNnCMTGTVFF is set. Thus the sequence of receptions can not be recovered completely now.

a) If FCNnCMTGTVFF is set, FCNnCMTGSSPM is no longer cleared on message transmission, but FCNnCMTGSSPM is still set, if all entries of FCNnCMTGTX are read by software.
b) i = 15 on 64 message buffer FCN;
i = 31 on 128 message buffer FCN.

Note Transmission from message buffers ...

- 0 to 16 (for 64 message buffer FCN)
- 0 to 32 (for 128 message buffer FCN)

... is not recorded to the transmit history list in the normal operation mode with ABT.

(b) FCNnCMTGTX write

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FCNnCMTGCLTV

FCNnCMTGCLTV	Setting of FCNnCMTGTVFF bit
0	FCNnCMTGTVFF bit is not changed
1	FCNnCMTGTVFF bit is cleared to 0

(14) FCNnCMTSCTL - FCNn module time stamp register

This register is used to control the time stamp function.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + 0 8298_H

Initial Value 0000_H. The register is initialized by any reset.

(a) FCNnCMTSCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	FCNnCMTSLOKE	FCNnCMTSSELE	FCNnCMTSTSGE

Note The lock function of the time stamp function must not be used when the FCN module is in the normal operation mode with ABT.

FCNnCMTSLOKE	Time stamp lock function enable bit
0	Time stamp lock function stopped. The TSOUT signal is toggled each time the selected time stamp capture event occurs.
1	Time stamp lock function enabled. The TSOUT signal is toggled each time the selected time stamp capture event occurs. However, the TSOUT output signal is locked when a data frame has been correctly received to message buffer 0 [a].

a) FCNnCMTSTSGE is automatically cleared to 0.

FCNnCMTSSELE	Time stamp capture event selection bit
0	The timestamp capture event is SOF.
1	The time stamp capture event is the last bit of EOF.

FCNnCMTSTSGE	TSOUT operation setting bit
0	TSOUT toggle operation is disabled.
1	TSOUT toggle operation is enabled.

(b) FCNnCMTSCTL write

15	14	13	12	11	10	9	8
0	0	0	0	0	FCNnCMTSSELK	FCNnCMTSSES	FCNnCMTSSETS
7	6	5	4	3	2	1	0
0	0	0	0	0	FCNnCMTSCLLK	FCNnCMTSCLSL	FCNnCMTSCLTS

FCNnCMTSSELK	FCNnCMTSCLLK	Setting of FCNnCMTSLOKE bit
0	1	FCNnCMTSLOKE is cleared to 0.
1	0	FCNnCMTSLOKE is set to 1.
Other than above		FCNnCMTSLOKE is not changed.

FCNnCMTSSESL	FCNnCMTSCLSL	Setting of FCNnCMTSSELE bit
0	1	FCNnCMTSSELE is cleared to 0.
1	0	FCNnCMTSSELE is set to 1.
Other than above		FCNnCMTSSELE is not changed.

FCNnCMTSSETS	FCNnCMTSCLTS	Setting of FCNnCMTSTSGE bit
0	1	FCNnCMTSTSGE is cleared to 0.
1	0	FCNnCMTSTSGE is set to 1.
Other than above		FCNnCMTSTSGE is not changed.

22.6.3 FCN message buffer registers

(1) FCNnMmDATxB/H/W, FCNn message data byte registers

These registers are used to store the data of a transmit/receive message.

Access The FCNnMmDATxW registers can be read/written in 32-bit units.
The FCNnMmDATxH registers can be read/written in 16-bit units.
The FCNnMmDATxB registers can be read/written in 8-bit units.

Address FCNnMmDAT0B: <FCNn_base> + 0 1000_H + m x 40_H
FCNnMmDAT1B: <FCNn_base> + 0 1004_H + m x 40_H
FCNnMmDAT2B: <FCNn_base> + 0 1008_H + m x 40_H
FCNnMmDAT3B: <FCNn_base> + 0 $100C_H$ + m x 40_H
FCNnMmDAT4B: <FCNn_base> + 0 1010_H + m x 40_H
FCNnMmDAT5B: <FCNn_base> + 0 1014_H + m x 40_H
FCNnMmDAT6B: <FCNn_base> + 0 1018_H + m x 40_H
FCNnMmDAT7B: <FCNn_base> + 0 $101C_H$ + m x 40_H

FCNnMmDAT0H: <FCNn_base> + 0 9000_H + m x 40_H
FCNnMmDAT2H: <FCNn_base> + 0 9008_H + m x 40_H
FCNnMmDAT4H: <FCNn_base> + 0 9010_H + m x 40_H
FCNnMmDAT6H: <FCNn_base> + 0 9018_H + m x 40_H

FCNnMmDAT0W: <FCNn_base> + 1 1000_H + m x 40_H
FCNnMmDAT4W: <FCNn_base> + 1 1010_H + m x 40_H

<R> **Initial Value** 0000 0000_H for FCNnMmDATxW. This register is initialized by any reset.
0000_H for FCNnMmDATxH. This register is initialized by any reset.
00_H for FCNnMmDATxB. This register is initialized by any reset.

(a) FCNnCMmDATxB (x = 0 to 7)

7	6	5	4	3	2	1	0
FCNnMmSSD0, FCNnMmSSD1, FCNnMmSSD2, FCNnMmSSD3, FCNnMmSSD4, FCNnMmSSD5, FCNnMmSSD6, FCNnMmSSD7							

(b) FCNnCMmDATxH (x = 0, 2, 4, 6)

15	14	13	12	11	10	9	8
FCNnMmSSD0, FCNnMmSSD2, FCNnMmSSD4, FCNnMmSSD6							

7	6	5	4	3	2	1	0
FCNnMmSSD1, FCNnMmSSD3, FCNnMmSSD5, FCNnMmSSD7							

(c) FCNnCMmDATxW (x = 0, 4)

31	30	29	28	27	26	25	24
FCNnMmSSD0, FCNnMmSSD4							

23	22	21	20	19	18	17	16
FCNnMmSSD1, FCNnMmSSD5							

15	14	13	12	11	10	9	8
FCNnMmSSD2, FCNnMmSSD6							

7	6	5	4	3	2	1	0
FCNnMmSSD3, FCNnMmSSD7							

(2) FCNnMmDTLGB - FCNn message data length register m

This register is used to set the number of bytes of the data field of a message buffer (DLC).

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + 0 1020_H + m x 40_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	FCNnMmDTLG[3:0]			

<R>

FCNnMmDTLG[3:0]	Data length of transmit/receive message
0000_B	0 bytes
0001_B	1 byte
0010_B	2 bytes
0011_B	3 bytes
0100_B	4 bytes
0101_B	5 bytes
0110_B	6 bytes
0111_B	7 bytes
1000_B	8 bytes
1001_B	Setting prohibited (If these bits are set during transmission, 8-byte data is transmitted regardless of the set FCNnMmDTLG[3:0] value when a data frame is transmitted. However, the DLC actually transmitted to the CAN bus is the DLC value set to this register.)Note
1010_B	
1011_B	
1100_B	
1101_B	
1110_B	
1111_B	

<R>

Note The data and DLC value actually transmitted to CAN bus are as follows.

Type of transmit frame	Length of transmit data	DLC transmitted
Data frame	Number of bits specified by FCNnMmDTLG[3:0] (However, 8 bytes if value ≥ 8)	FCNnMmDTLGB.FCNnMmDTLG[3:0] bits
Remote frame	0 bytes	

Cautions

1. Be sure to set bits 7 to 4 to 0000_B.
2. Receive data is stored in as many FCNnMmDATxB register as the number of bytes (however, the upper limit is 8) corresponding to DLC of the received frame. The FCNnMmDATxB register in which no data is stored is undefined.
3. On reception, FCNnMmDTLGB is updated according to the received frame.

(3) FCNnMmSTRB - FCNn message configuration register m

This register is used to specify the type of the message buffer and to set a mask.

Access This register can be read/written in 8-bit units.

<R> **Address** <FCNn_base> + 0 1024_H + m x 40_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
FCNnMm SSOW	FCNnMm SSMT[3:0]				FCNnMm SSRT	0	FCNnMm SSAM

FCNnMmSSOW	Overwrite control bit
0	The message buffer that has already received a data frame[a] is not overwritten by a newly received data frame. The newly received data frame is discarded.
1	The message buffer that has already received a data frame[a] is overwritten by a newly received data frame.

a) The “message buffer that has already received a data frame” is a receive message buffer whose FCNnMmCTL.FCNnMmDTNF bit has been set to 1.

Note A remote frame is received and stored, regardless of the setting of FCNnMmCTL.FCNnMmSSOW and FCNnMmCTL.FCNnMmDTNF. A remote frame that satisfies the other conditions is always received and stored in the corresponding message buffer (interrupt generated, FCNnMmDTNF flag set, FCNnMmDTLGB.FCNnMmDTLG[3:0] updated, and recorded to the receive history list).

FCNnMmSSRT	Remote frame request bit
0	Transmit / receive a data frame.
1	Transmit / receive a remote frame.

FCNnMmSTRB.FCNnMmSSRT specifies the type of message frame that is transmitted or received from/to a message buffer.

Notes

1. If the message buffer is defined as a transmit message buffer, and a remote frame shall be received into it, the FCNnMmSSRT bit must be cleared.
2. Even if a valid remote frame has been received in a transmit message buffer, the FCNnMmSSRT bit of the transmit message buffer that has received the frame remains cleared to 0.
3. Even if a remote frame whose ID matches has been received from the CAN bus, if the FCNnMmSSRT bit of a transmit message buffer is set to 1 (to transmit a remote frame), that remote frame is not stored in this transmit message buffer.
4. If the message buffer is defined as a receive message buffer, the FCNnMmSSRT bit must be set, in order to receive remote frames instead of data frames.

FCNnMmSSMT[3:0]	Message buffer type setting bit
0000_B	Transmit message buffer
0001_B	Receive message buffer (no mask setting)
0010_B	Receive message buffer (mask 1 set)
0011_B	Receive message buffer (mask 2 set)
0100_B	Receive message buffer (mask 3 set)
0101_B	Receive message buffer (mask 4 set)
0110_B	Receive message buffer (mask 5 set)
0111_B	Receive message buffer (mask 6 set)
1000_B	Receive message buffer (mask 7 set)
1001_B	Receive message buffer (mask 8 set)
Other than above	Setting prohibited

Note The setting of FCNnMmSSMT is also valid to select masks in conjunction with remote frame reception. To receive remote frames in receive message buffers, the flag FCNnMmSSRT of the message buffer must be set.

FCNnMmSSAM	Message buffer assignment bit
0	Message buffer not used.
1	Message buffer used.

Caution Be sure to write 0 to bit 1.

(4) FCNnMmMID0H, FCNnMmMID1H, FCNnMmMID0W - FCNn message ID register m

These registers are used to set an identifier (ID).

Access FCNnMmMID0H, FCNnMmMID1H can be read/written in 16-bit units.
FCNnMmMID0W can be read/written in 32-bit units.

<R> **Address** FCNnMmMID0H: <FCNn_base> + 0 9028_H + m x 40_H
FCNnMmMID1H: <FCNn_base> + 0 9030_H + m x 40_H

FCNnMmMID0W: <FCNn_base> + 1 1028_H + m x 40_H

<R> **Initial Value** 0000_H for FCNnMmMID0H and FCNnMmMID1H. This register is initialized by any reset.
0000 0000_H for FCNnMmMID0W. This register is initialized by any reset.

(a) FCNnMmMID0H

15	14	13	12	11	10	9	8
FCNnMmSSID[15:8]							

7	6	5	4	3	2	1	0
FCNnMmSSID[7:0]							

(b) FCNnMmMID1H

15	14	13	12	11	10	9	8
FCNnMmSSIE	0	0	FCNnMmSSID[28:24]				

7	6	5	4	3	2	1	0
FCNnMmSSID[23:16]							

(c) FCNnCMmMID0W

31	30	29	28	27	26	25	24
FCNnMmSSIE	0	0	FCNnMmSSID[28:24]				

23	22	21	20	19	18	17	16
FCNnMmSSID[23:16]							

15	14	13	12	11	10	9	8
FCNnMmSSID[15:8]							

7	6	5	4	3	2	1	0
FCNnMmSSID[7:0]							

FCNnMmSSIE	Format mode specification bit
0	Standard format mode (FCNnMmSSID[28:18]: 11 bits, FCNnMmSSID[17:0] are not used)
1	Extended format mode (FCNnMmSSID[28:0]: 29 bits)

FCNnMmSSID[28:0]	Message ID
FCNnMmSSID[28:18]	Standard ID value of 11 bits (when FCNnMmSSIE = 0)
FCNnMmSSID[28:0]	Extended ID value of 29 bits (when FCNnMmSSIE = 1)

Cautions

1. Be sure to write 0 to bits 14 and 13 of FCNnMmMID1H, respectively bits 30 and 29 of FCNnMmMID0W register.
2. Be sure to align the ID value according to the given bit positions into this registers. Note that for standard ID, the ID value must be shifted to fit into FCNnMmSSID[28:18] bit positions.

(5) FCNnMmCTL - FCNn message control register m

This register is used to control the operation of the message buffer.

Access This register can be read/written in 16-bit units.

<R> **Address** <FCNn_base> + 0 9038_H + m x 40_H

Initial Value 0000_H. This register is initialized by any reset.

(a) FCNnMmCTL read

15	14	13	12	11	10	9	8
0	0	FCNnMm MUCF	0	0	0	FCNnMm TCPF	0
7	6	5	4	3	2	1	0
0	FCNnMm NHMF	0	FCNnMm MOWF	FCNnMm IENF	FCNnMm DTNF	FCNnMm TRQF	FCNnMm RDYF

<R>

FCNnMmMUCF	Bit indicating that message buffer data is being updated
0	The FCN module is not updating the message buffer (no data is being received and stored).
1	The FCN module is updating the message buffer (data is being received and stored).

FCNnMmTCPF[a]	Transmission complete flag
0	Transmission failed.[b]
1	Transmission is complete.

a) FCNnMmTCPF is cleared if FCNnMmRDYF is changed or FCNnMmTRQF is set.

b) If transmission abort was requested by clearing the FCNnMmTRQF flag by the application, FCNnMmTCPF = 0 indicates a successful transmission abort.

FCNnMmNHMF	History mask flag [a]
0	Updating of the receive history list register FCNnCMRGRX and transmit history list register FCNnCMTGTX is not masked.
1	Updating of the receive history list register FCNnCMRGRX and transmit history list register FCNnCMTGTX is masked.

a) If updating is masked, the transmit and receive history lists are not updated even when reception or transmission on the corresponding message buffer finishes.

FCNnMmMOWF	Message buffer overwrite status bit
0	The message buffer is not overwritten by a newly received data or remote frame.
1	The message buffer is overwritten by a newly received data or remote frame.

Note This bit will not be set (1) if a remote frame is received and stored in a transmit message buffer with FCNnMmDTNF = 1.

FCNnMmIENF	Message buffer interrupt request enable bit
0	Receive message buffer: Valid message reception completion interrupt disabled. Transmit message buffer: Normal message transmission completion interrupt and transmit abort interrupt disabled.
1	Receive message buffer: Valid message reception completion interrupt enabled. Transmit message buffer: Normal message transmission completion interrupt enabled.

Caution Set FCNnMmIENF and FCNnMmRDYF always separately.

FCNnMmDTNF	Message buffer data update bit
0	A new data frame or remote frame has been stored in the message buffer.
1	No new data frame or remote frame has been stored in the message buffer.

Caution Do not set FCNnMmDTNF to 1 by software. Be sure to write 0 to bit 10.

FCNnMmTRQF	Message buffer transmission request bit
0	No message frame transmitting request that is pending or being transmitted is in the message buffer.
1	The message buffer is holding transmission of a message frame pending or is transmitting a message frame.

Cautions
1. Do not set FCNnMmTRQF and FCNnMmRDYF to 1 at the same time. Set FCNnMmRDYF = 1 before setting FCNnMmTRQF = 1.
2. Only set FCNnMmTRQF to 1 for transmit message buffers (not to buffers for which FCNnMmSSMT[3:0] ≠ 4'b0000 or FCNnMmSSAM = 0).

FCNnMmRDYF	Message buffer ready bit
0	The message buffer can be written by software. The FCN module cannot write to the message buffer.
1	Writing the message buffer by software is ignored (except a write access to the FCNnMmRDYF, FCNnMmTRQF, FCNnMmDTNF, and FCNnMmMOWF). The FCN module can write to the message buffer.

Cautions

1. Set FCNnMmIENF and FCNnMmRDYF always separately.
2. Do not set FCNnMmTRQF and FCNnMmRDYF to 1 at the same time. Set FCNnMmRDYF = 1 before setting FCNnMmTRQF = 1.
3. Do not clear FCNnMmRDYF to “0” during message transmission. Follow the transmission abort process about clearing FCNnMmRDYF for redefinition of the message buffer.
4. Clearing of FCNnMmRDYF may take some time, depending on activity of the CAN Controller. Repeat the clearing access, until reading of FCNnMmRDYF confirms that the bit is cleared.
5. Be sure that FCNnMmRDYF is cleared before writing to the other message buffer registers, by checking the status of FCNnMmRDYF.

(b) FCNnMmCTL write

15	14	13	12	11	10	9	8
0	FCNnMm SENH	0	0	FCNnMm SEIE	0	FCNnMm SETR	FCNnMm SERY
7	6	5	4	3	2	1	0
0	FCNnMm CLNH	0	FCNnMm CLMW	FCNnMm CLIE	FCNnMm CLDN	FCNnMm CLTR	FCNnMm CLRY

FCNnMmSENH	FCNnMmCLNH	Setting of FCNnMmNHMF bit
0	1	FCNnMmNHMF is cleared.
1	0	FCNnMmNHMF is set.
Other than above		FCNnMmNHMF is not changed.

FCNnMmCLMW	Setting of FCNnMmMOWF bit
0	FCNnMmMOWF is not changed.
1	FCNnMmMOWF is cleared.

FCNnMmSEIE	FCNnMmCLIE	Setting of FCNnMmIENF bit
0	1	FCNnMmIENF is cleared.
1	0	FCNnMmIENF is set.
Other than above		FCNnMmIENF is not changed.

FCNnMmCLDN	Setting of FCNnMmDTNF bit
1	FCNnMmDTNF is cleared.
0	FCNnMmDTNF is set.

Note If FCNnMmDTNF is cleared at the end of ID field reception, the frames being received will be saved into the corresponding message buffer.

FCNnMmSETR	FCNnMmCLTR	Setting of FCNnMmTRQF bit
0	1	FCNnMmTRQF is cleared.
1	0	FCNnMmTRQF is set.
Other than above		FCNnMmTRQF is not changed.

FCNnMmSERY	FCNnMmCLRY	Setting of FCNnMmRDYF bit
0	1	FCNnMmRDYF is cleared.
1	0	FCNnMmRDYF is set.
Other than above		FCNnMmRDYF is not changed.

22.7 CAN Controller Initialization

22.7.1 Initialization of FCN module

Before FCN module operation is enabled, the FCN module system clock needs to be determined by setting FCNnGMCSPRE.FCNnGMCSPRSC[3:0] by software. Do not change the setting of the FCN module system clock after FCN module operation is enabled.

The FCN module is enabled by setting FCNnGMCLCTL.FCNnGMCLPWOM.

For the procedure of initializing the FCN module, refer to *22.15 “Operation of the CAN Controller” on page 1570*.

22.7.2 Initialization of message buffer

After the FCN module is enabled, the message buffers might contain an undefined value (except after a software reset). A minimum initialization for all the message buffers, even for those not used in the application, is necessary before switching the FCN module from the initialization mode to one of the operation modes.

- Clear FCNnMmRDYF, FCNnMmTRQF, and FCNnMmDTNF of the FCNnMmCTL registers to 0.
- Clear all FCNnMmSTRB.FCNnMmSSAM to 0.

22.7.3 Redefinition of message buffer

Redefining a message buffer means changing the ID and control information of the message buffer while a message is being received or transmitted, without affecting other transmission/reception operations.

(1) To redefine message buffer in initialization mode

Place the FCN module in the initialization mode once and then change the ID and control information of the message buffer in the initialization mode. After changing the ID and control information, set the FCN module to an operation mode.

(2) To redefine message buffer during reception

Perform redefinition as shown in *Figure 22-18 “Message buffer redefinition during reception” on page 1574*.

(3) To redefine message buffer during transmission

To rewrite the contents of a transmit message buffer to which a transmission request has been set, perform transmission abort processing (see *(1) “Transmission abort process except for in normal operation mode with automatic block transmission (ABT)”* and *(2) “Transmission abort process for ABT transmission in normal operation mode with automatic block transmission (ABT)” on page 1548* for details). Confirm that transmission has been aborted or completed, and then redefine the message buffer. After redefining the transmit message buffer, set a transmission request using the procedure described below. When setting a transmission request to a message buffer that

has been redefined without aborting the transmission in progress, however, the 1-bit wait time is not necessary.

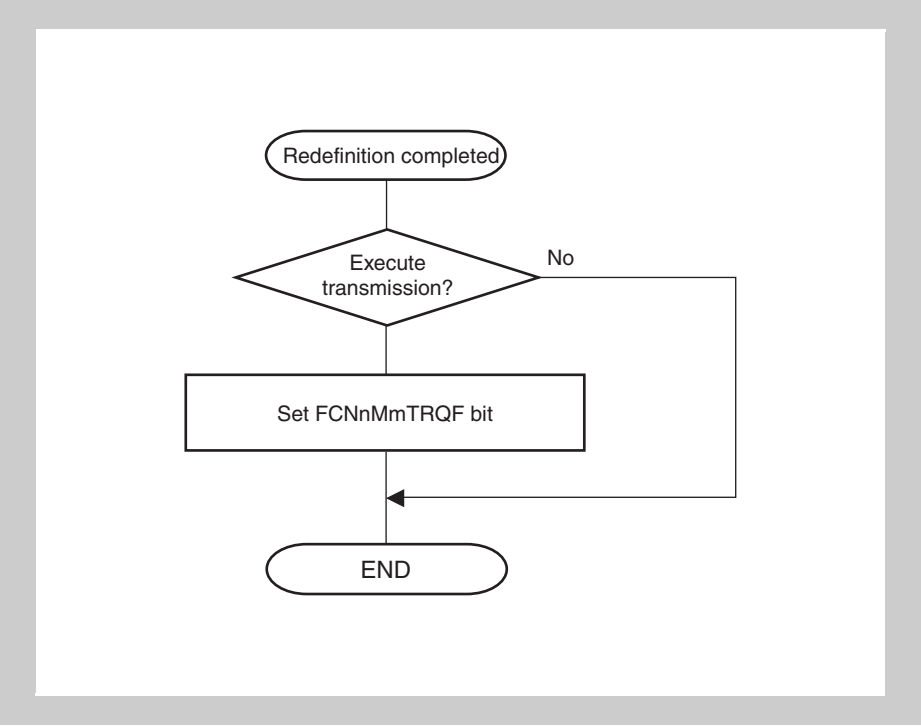


Figure 22-5 Setting transmission request (FCNnMmCTL.FCNnMmTRQF) to transmit message buffer after redefinition

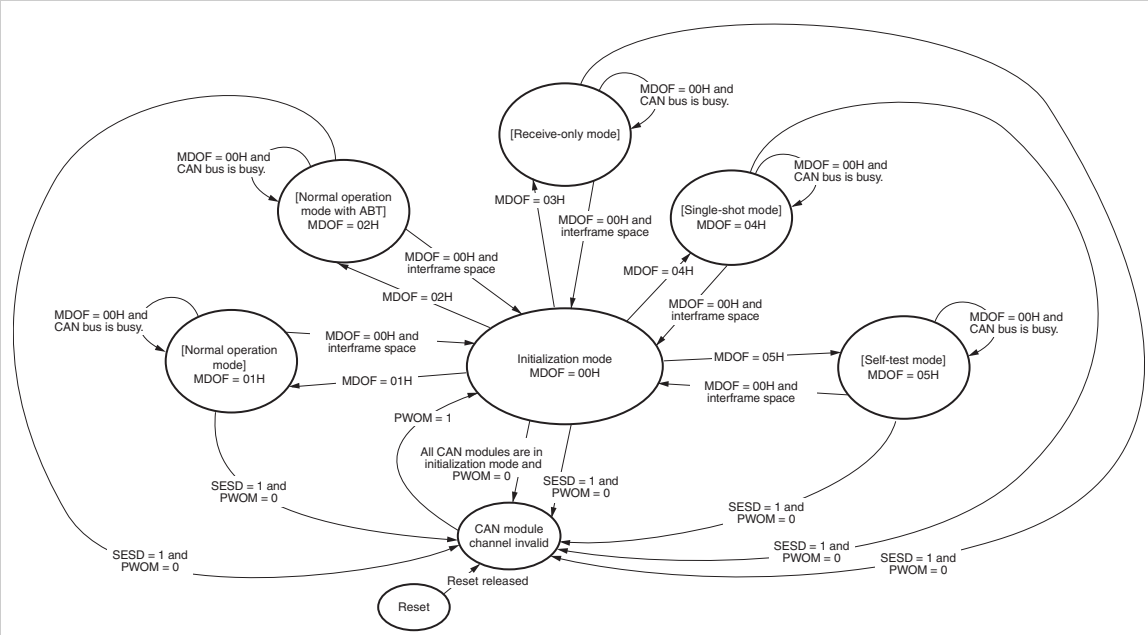
Cautions

1. When a message is received, reception filtering is performed in accordance with the ID and mask set to each receive message buffer. If the procedure in *Figure 22-18 "Message buffer redefinition during reception" on page 1574* is not observed, the contents of the message buffer after it has been redefined may contradict the result of reception (result of reception filtering). If this happens, check that the ID and IDE received first and stored in the message buffer following redefinition are those stored after the message buffer has been redefined. If no ID and IDE are stored after redefinition, redefine the message buffer again.
2. When a message is transmitted, the transmission priority is checked in accordance with the ID, IDE, and FCNnMmSTRB.FCNnMmSSRT set to each transmit message buffer to which a transmission request was set. The transmit message buffer having the highest priority is selected for transmission. If the procedure in *Figure 22-5 "Setting transmission request (FCNnMmCTL.FCNnMmTRQF) to transmit message buffer after redefinition" on page 1531* is not observed, a message with an ID not having the highest priority may be transmitted after redefinition.

22.7.4 Transition from initialization mode to operation mode

The FCN module can be switched to the following operation modes.

- Normal operation mode
- Normal operation mode with ABT
- Receive-only mode
- Single-shot mode
- Self-test mode



<R> **Figure 22-6 Transition to operation modes**

Note In the figure above following abbreviations are used:

- MDOF = FCNnCMCLCTL.FCNnCMCLMDOF[2:0]
- PWOM = FCNnGMCLCTL.FCNnGMCLPWOM
- SESD = FCNnGMCLCTL.FCNnGMCLSESD

The transition from the initialization mode to an operation mode is controlled by the bit string FCNnCM.FCNnCMCLMDOF[2:0].

Changing from one operation mode into another requires shifting to the initialization mode in between. Do not change one operation mode to another directly; otherwise the operation will not be guaranteed.

Requests for transition from an operation mode to the initialization mode are held pending when the CAN bus is not in the interframe space (i.e., frame reception or transmission is in progress), and the FCN module enters the initialization mode at the first bit in the interframe space (the values of the FCNnCMCLCTL.FCNnCMCLMDOF[2:0] are changed to $000_{B)}$. After issuing a request to change the mode to the initialization mode, read FCNnCMCLCTL.FCNnCMCLMDOF[2:0] until their value becomes 000_B to confirm that the module has entered the initialization mode (see *Figure 22-15 “Re-initialization” on page 1571*).

<R>

22.8 Message Reception

22.8.1 Message reception

In all the operation modes, the complete message buffer area is analyzed to find a suitable buffer to store a newly received message. All message buffers satisfying the following conditions are included in that evaluation (RX-search process).

- Used as a message buffer
  (FCNnMmSTRB.FCNnMmSSAM = 1.)
- Set as a receive message buffer
  (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0001_B to 1001_B)
- Ready for reception
  (FCNnMmCTL.FCNnMmRDYF = 1.)

When two or more message buffers of the FCN module are found to be able to receive a message, the message is stored according to the priority explained below. The message is always stored in the message buffer with the highest priority, not in a message buffer with a low priority. For example, when an unmasked receive message buffer and a receive message buffer linked to mask 1 have the same ID, the received message is not stored in the message buffer linked to mask 1, even if that message buffer has not received a message and a message has already been received in the unmasked receive message buffer. In other words, when a condition has been set in two or more message buffers with different priorities, the message buffer with the highest priority always stores the message; the message is not stored in message buffers with a lower priority. This also applies when the message buffer with the highest priority is unable to store a message (i.e., when FCNnMmCTL.FCNnMmDTNF = 1 indicating that a message has already been received, but rewriting is disabled because FCNnMmSTRB.FCNnMmSSOW = 0). In this case, the message is not actually stored in the candidate message buffer with the highest priority, but neither is it stored in a message buffer with a lower priority.

Table 22-19 MBRB priorities

Priority	Storing condition if same ID is set	
1 (high)	Unmasked message buffer	FCNnMmDTNF = 0
		FCNnMmDTNF = 1 and FCNnMmSSOW = 1
2	Message buffer linked to mask 1	FCNnMmDTNF = 0
		FCNnMmDTNF = 1 and FCNnMmSSOW = 1
3	Message buffer linked to mask 2	FCNnMmDTNF = 0
		FCNnMmDTNF = 1 and FCNnMmSSOW = 1
...	...	
9 (low)	Message buffer linked to mask 8	FCNnMmDTNF = 0
		FCNnMmDTNF = 1 and FCNnMmSSOWt = 1

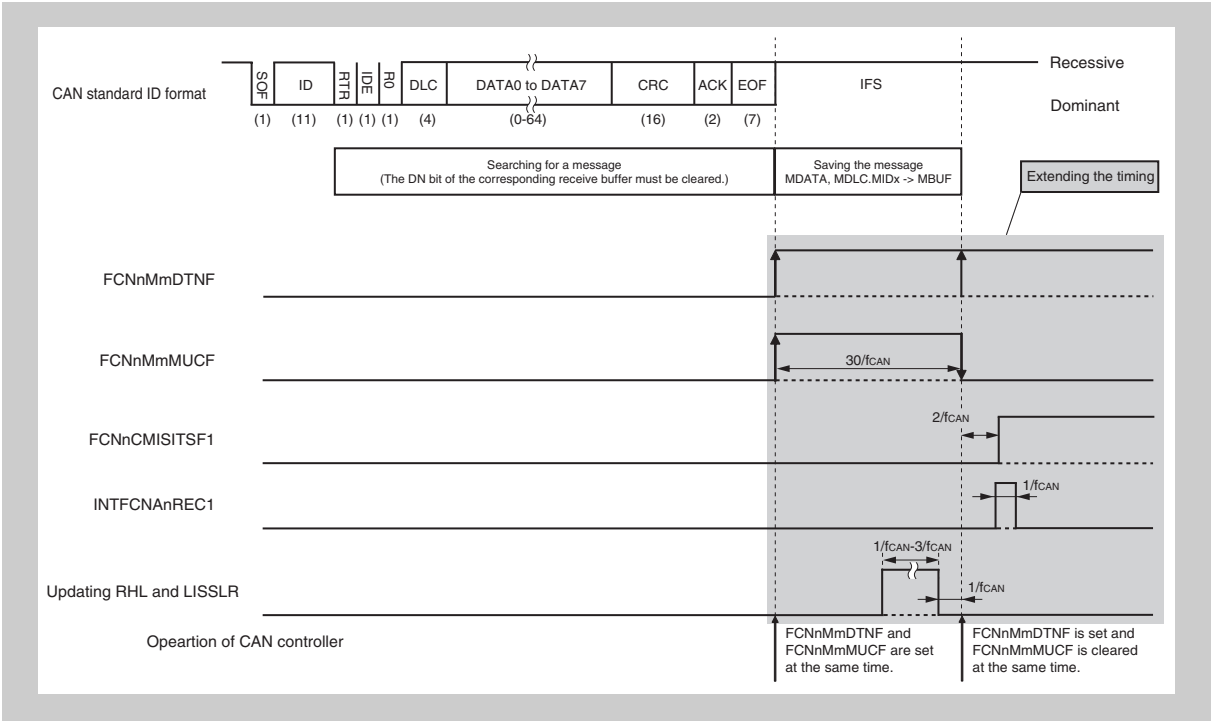
22.8.2 Receive data read

To keep data consistency when reading FCN message buffers, perform the data reading according to *Figure 22-32 “Reception via interrupt (using FCNnCMLISTR register)” on page 1589* to *Figure 22-35 “Reception via software polling” on page 1593*.

During message reception, the FCN module sets FCNnMmCTL.FCNnMmDTNF two times: at the beginning of the storage process of data to the message buffer, and again at the end of this storage process. During this storage process, FCNnMmCTL.FCNnMmMUCF of the message buffer is set (refer to *Figure 22-7 “Reception timing”*).

The receive history list is also updated just before the storage process. In addition, during storage process (FCNnMmCTL.FCNnMmMUCF = 1), FCNnMmCTL.FCNnMmRDYF of the message buffer is locked to avoid any coincidental data write by CPU. Note that the storage process may be disturbed (delayed) when the CPU accesses the message buffer.

<R> **Caution** To reliably store a message in a message buffer, the DN bit for that buffer must be cleared before message search processing starts (after a frame ID is output on the bus). This might occur as early as the 15th CAN bit after EOF of the previous frame. To reliably receive successively sent CAN frames on the bus, using multiple message buffers for frame reception is recommended.



<R> **Figure 22-7 Reception timing**

22.8.3 Receive history list function

The receive history list (RHL) function records in the receive history list the number of the receive message buffer in which each data frame or remote frame was received and stored. The RHL consists of storage elements equivalent to up to 47 messages (on 64 message buffer FCN) or up to 95 messages (on 128 message buffer FCN), the last in-message pointer FCNnCMLISSLR[7:0] with the corresponding FCNnCMLISTR register and the receive history list get pointer FCNnCMRGSSPT with the corresponding FCNnCMRGRX register.

The RHL is undefined immediately after the transition of the FCN module from the initialization mode to one of the operation modes.

The FCNnCMLISTR register holds the contents of the RHL element indicated by the value of the FCNnCMLISTR.FCNnCMLISSLR[7:0] pointer minus 1. It is consequently possible to check the number of the message buffer that received and stored the last data frame or remote frame by reading the FCNnCMLISTR register. The FCNnCMLISSLR[7:0] pointer is utilized as a write pointer that indicates to what part of the RHL a message buffer number is recorded. Any time a data frame or remote frame is received and stored, the corresponding message buffer number is recorded to the RHL element indicated by the FCNnCMLISSLR[7:0] pointer. Each time recording to the RHL has been completed, the FCNnCMLISSLR[7:0] pointer is automatically incremented. In this way, the number of the message buffer that has received and stored a frame will be recorded chronologically.

For message buffers, where the flag FCNnMmCTL.FCNnMmNHMF is set, no entry in the history lists is recorded.

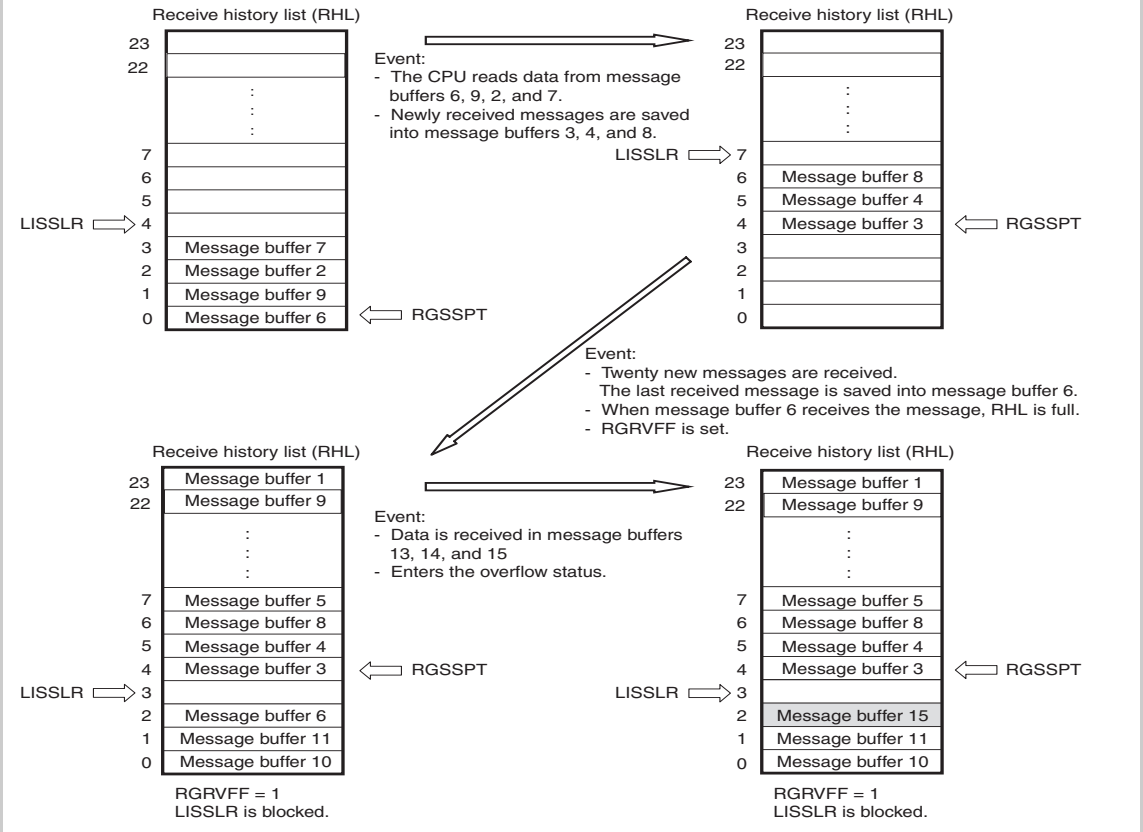
The FCNnCMRGRX.FCNnCMRGSSPT pointer is utilized as a read pointer that reads a recorded message buffer number from the RHL. This pointer indicates the first RHL element that the CPU has not read yet. By reading the FCNnCMRGRX register by software, the number of a message buffer that has received and stored a data frame or remote frame can be read. Each time a message buffer number is read from the FCNnCMRGRX register, the FCNnCMRGSSPT pointer is automatically incremented.

If the value of the FCNnCMRGRX.FCNnCMRGSSPT pointer matches the value of the FCNnCMLISTR.FCNnCMLISSLR[7:0] pointer, FCNnCMRGRX.FCNnCMRGSSPM (receive history list pointer match) is set to 1. This indicates that no message buffer number that has not been read remains in the RHL. If a new message buffer number is recorded, the FCNnCMLISSLR[7:0] pointer is incremented and because its value no longer matches the value of the FCNnCMRGSSPT pointer, FCNnCMRGSSPM is cleared. In other words, the numbers of the unread message buffers exist in the RHL.

If the FCNnCMLISTR.FCNnCMLISSLR[7:0] pointer is incremented and matches the value of the FCNnCMRGRX.FCNnCMRGSSPT pointer minus 1, FCNnCMRGRX.FCNnCMRGRVFF (receive history list overflow) is set to 1. This indicates that the RHL is full of numbers of message buffers that have not been read. When further message reception and storing occur, the last recorded message buffer number is overwritten by the number of the message buffer that received and stored the newly received message. In this case, after FCNnCMRGRVFF has been set (1), the recorded message buffer numbers in the RHL do not completely reflect the chronological order. However messages itself are not lost and can be located by CPU search in message buffer memory with the help of FCNnMmCTL.FCNnMmDTNF, or by reading the global registers FCNnDNBMRX[3:0].

Caution If the history list is in the overflow condition (FCNnCMRGRX.FCNnCMRGRVFF is set), reading the history list contents is still possible, until the history list is empty (indicated by FCNnCMRGRX.FCNnCMRGSSPM flag set). Nevertheless, the history list remains in the overflow condition, until FCNnCMRGRVFF is cleared by software. If FCNnCMRGRVFF is not cleared, the FCNnCMRGSSPM flag will also not be updated (cleared) upon a message storage of newly received frame. This may lead to the situation, that FCNnCMRGSSPM indicates an empty history list, although a reception has taken place, while the history list is in the overflow state (FCNnCMRGRVFF and FCNnCMRGSSPM are set).

As long as the RHL still has free entries, the sequence of occurrence is maintained. If more receptions occur without reading the RHL by the host processor, complete sequence of receptions can not be recovered.



RGRVFF = 1 denoted that LISSLR equals RGSSPT - 1 while message buffer number stored to element indicated by LISSLR - 1.

Note: 1. The following abbreviations are used in the figure above.
- LISSLR = Last reception message pointer FCNnCMLISTR.FCNnCMLISSLR[7:0]
- RGSSPT = Reception history list acquisition pointer FCNnCMRGRX.FCNnCMRGSSPT[7:0]
- RGRVFF = Reception history list overflow flag FCNnCMRGRX.FCNnCMRGRVFF

2. If a message buffer with the FCNnMmCTL.FCNnMmNHMF flag set receives a message, the message buffer number indicated by LISSLR and the message buffer number written to RHL will differ.

Figure 22-8 Receive history list

22.8.4 Mask function

For any message buffer, which is used for reception, the assignment to one of eight global reception masks (or no mask) can be selected.

By using the mask function, the message ID comparison can be reduced by masked bits, herewith allowing the reception of several different IDs into one buffer.

While the mask function is in effect, an identifier bit that is defined to be 1 by a mask in the received message is not compared with the corresponding identifier bit in the message buffer.

However, this comparison is performed for any bit whose value is defined as 0 by the mask.

For example, let us assume that all messages that have a standard-format ID, in which bits ID27 to ID25 are 0 and bits ID24 and ID22 are 1, are to be stored in message buffer 14. The procedure for this example is shown below.

(1) Identifier to be stored in message buffer

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x

(2) Identifier to be configured in message buffer 14 (example) (using FCNnM014MID0W register)

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x
ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
x	x	x	x	x	x	x	x	x	x	x
ID6	ID5	ID4	ID3	ID2	ID1	ID0				
x	x	x	x	x	x	x				

Notes

1. ID with the ID27 to ID25 bits cleared to 0 and the ID24 and ID22 bits set to 1 is registered (initialized) to message buffer 14.
2. Message buffer 14 is set as a standard format identifier that is linked to mask 1 (FCN1M014STRB.FCN1M014SSMT[3:0] = 0010_B).

(3) Mask setting for FCN module 1 (mask 1) (example) (using FCAN1 module mask 1 register FCN1CMMKCTL01W)

FNCnCMMKSSID[..]

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x
ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
x	x	x	x	x	x	x	x	x	x	x
ID6	ID5	ID4	ID3	ID2	ID1	ID0				
x	x	x	x	x	x	x				

1: Not compared (masked)
0: Compared

FCN1CMMKSSID[27:24] and FCN1CMMKSSID[21] are cleared to 0, and FCN1CMMKSSID[28], FCN1CMMKSSID[23], and FCN1CMMKSSID[21:0] are set to 1.

22.8.5 Multi buffer receive block function

The multi buffer receive block (MBRB) function is used to store a block of data in two or more message buffers sequentially with no CPU interaction, by setting the same ID to two or more message buffers with the same message buffer type. These message buffers can be allocated anywhere in the message buffer memory, they do not even have to follow each other adjacently.

Suppose, for example, the same message buffer type is set to 10 message buffers, message buffers 10 to 19, and the same ID is set to each message buffer. If the first message whose ID matches an ID of the message buffers is received, it is stored in message buffer 10. At this point, FCNnMmCTL.FCNnMmDTNF of message buffer 10 is set, prohibiting overwriting the message buffer when subsequent messages are received.

When the next message with a matching ID is received, it is received and stored in message buffer 11. Each time a message with a matching ID is received, it is sequentially (in the ascending order) stored in message buffers 12, 13, and so on. Even when a data block consisting of multiple messages is received, the messages can be stored and received without overwriting the previously received matching-ID data.

Whether a data block has been received and stored can be checked by setting FCNnMmCTL.FCNnMmIENF of each message buffer. For example, if a data block consists of k messages, k message buffers are initialized for reception of the data block. FCNnMmIENF in message buffers 0 to (k-2) is cleared to 0 (interrupts disabled), and FCNnMmIENF in message buffer k-1 is set to 1 (interrupts enabled). In this case, a reception completion interrupt occurs when a message has been received and stored in message buffer k-1, indicating that MBRB has become full. Alternatively, by clearing FCNnMmIENF of message buffers 0 to (k-3) and setting FCNnMmIENF of message buffer k-2, a warning that MBRB is about to overflow can be issued.

The basic conditions of storing receive data in each message buffer for the MBRB are the same as the conditions of storing data in a single message buffer.

Cautions

1. MBRB can be configured for each of the same message buffer types. Therefore, even if a message buffer of another MBRB whose ID matches but whose message buffer type is different has a vacancy, the received message is not stored in that message buffer, but instead discarded.
2. MBRB does not have a ring buffer structure. Therefore, after a message is stored in the message buffer having the highest number in the MBRB configuration, a newly received message will not be stored in the message buffer having the lowest message buffer number.
3. MBRB operates based on the reception and storage conditions; there are no settings dedicated to MBRB, such as function enable bits. By setting the same message buffer type and ID to two or more message buffers, MBRB is automatically configured.
4. With MBRB, “matching ID” means “matching ID after mask”. Even if the ID set to each message buffer is not the same, if the ID that is masked by the mask register matches, it is considered a matching ID and the buffer that has this ID is treated as the storage destination of a message.
5. The priority between MBRBs is mentioned in the table *Table 22-19 “MBRB priorities”*.

22.8.6 Remote frame reception

In all the operation modes, when a remote frame is received, the message buffer that is to store the remote frame is searched from all the message buffers satisfying the following conditions (1 and 2, condition 1 has priority on reception acceptance). If condition 1 is not fulfilled, the remaining message buffers are scanned, whether condition 2 could be fulfilled.

- Condition 1:
  Set as a transmit message buffer
  (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B)
  - Used as a message buffer
    (FCNnMmSTRB.FCNnMmSSAM = 1.)
  - Ready for reception
    (FCNnMmCTL.FCNnMmRDYF = 1.)
  - Set to data frame message type
    (FCNnMmSTRB.FCNnMmSSRT = 0.)
  - Transmission request is not set.
    (FCNnMmCTL.FCNnMmTRQF = 0.)
- Condition 2:
  Set as a receive message buffer
  (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0001_B ... 1001_B)
  - Used as a message buffer
    (FCNnMmSTRB.FCNnMmSSAM = 1.)
  - Ready for reception
    (FCNnMmCTL.FCNnMmRDYF = 1.)
  - Set to remote frame message type
    (FCNnMmSTRB.FCNnMmSSRT = 1.)
  - Buffer is ready to store a message
    (FCNnMmCTL.FCNnMmDTNF = 0, or FCNnMmSTRB.FCNnMmSSOW = 1 with FCNnMmCTL.FCNnMmDTNF = 1).

Upon acceptance of a remote frame, the following actions are executed if the ID of the received remote frame matches the ID of a message buffer that satisfies the above conditions.

- The FCNnMmDTLG[3:0] bit string in the FCNnMmDTLGB register store the received DLC value.
- When received in a transmit message buffer, registers FCNnMmDAT0B to FCNnMmDAT7B in the data area will not be updated (the data from before reception is stored).
- FCNnMmCTL.FCNnMmDTNF is set to 1.
- FCNnCMISCTL.FCNnCMISITSF1 is set to 1 (if FCNnMmCTL.FCNnMmIENF of the message buffer that receives and stores the frame is set to 1).
- The receive completion interrupt (INTCnREC) is output (if FCNnMmCTL. FCNnMmIENF of the message buffer that receives and stores the frame is set to 1 and if FCNnCMIECTL.FCNnCMIESEIE1 is set to 1).
- The message buffer number is recorded in the receive history list, if the flag FCNnMmCTL.FCNnMmNHMF is not set.

Caution When a transmit message buffer is found for receiving and storing a remote frame, overwrite control by FCNnMmSTRB.FCNnMmSSOW of the message buffer and FCNnMmCTL.FCNnMmDTNF are not checked. The setting of FCNnMmSSOW is ignored, and FCNnMmDTNF is set in any case.

Notes

1. If more than one transmit message buffer has the same ID and the ID of the received remote frame matches that ID, the remote frame is stored in the transmit message buffer with the lowest message buffer number.
2. If transmit and receive message buffers are found, which could receive a remote frame matching with its ID, either masked or unmasked, the remote frame is stored in the transmit message buffer.
3. If several receive message buffers would match for reception for a remote frame, the reception priority is identical as for a data frame.
4. If a receive message buffer is found to match for a remote frame reception, and selected for storage, but this receive message buffer does not allow the storage, because FCNnMmDTNF is set, and FCNnMmSSOW is not set, the remote frame is not stored at all.

22.9 Message Transmission

22.9.1 Message transmission

A message buffer with its FCNnMmCTL.FCNnMmTRQF bit set to 1 participates in the search for the most high-prioritized message when the following conditions are fulfilled. This behavior is valid for all operational modes.

- Used as a message buffer (FCNnMmSTRB.FCNnMmSSAM = 1)
- Set as a transmit message buffer (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B)
- Ready for transmission (FCNnMmCTL.FCNnMmRDYF = 1)

The CAN system is a multi-master communication system. In a system like this, the priority of message transmission is determined based on message identifiers (IDs). To facilitate transmission processing by software when there are several messages awaiting transmission, the FCN module uses hardware to check the ID of the message with the highest priority and automatically identifies that message. This eliminates the need for software-based priority control.

Transmission priority is controlled by the identifier (ID).

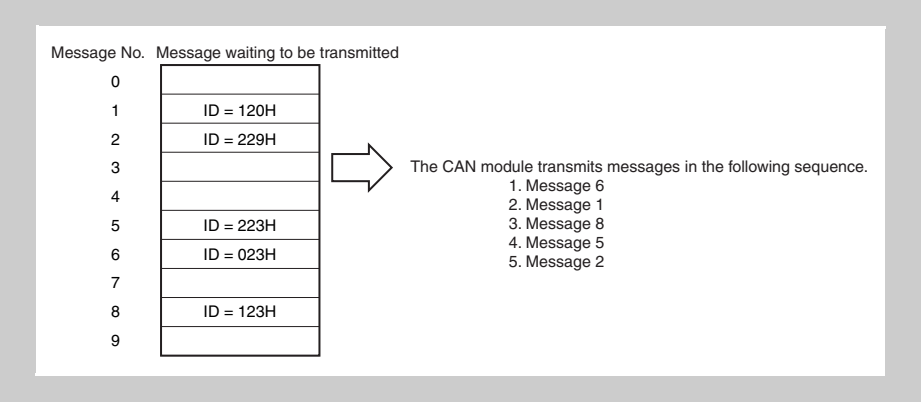


Figure 22-9 Message processing example

After the transmit message search, the transmit message with the highest priority of the transmit message buffers that have a pending transmission request (message buffers with the FCNnMmCTL.FCNnMmTRQF bit set to 1 in advance) is transmitted.

If a new transmission request is set, the transmit message buffer with the new transmission request is compared with the transmit message buffer with a pending transmission request. If the new transmission request has a higher priority, it is transmitted, unless transmission of a message with a low priority has already started. If transmission of a message with a low priority has already started, however, the new transmission request is transmitted later. To solve this priority inversion effect, the software can perform a transmission abort request for the lower priority message. The highest priority is determined according to the following rules.

Priority	Conditions	Description
1 (high)	Value of first 11 bits of ID [ID28 to ID18]:	The message frame with the lowest value represented by the first 11 bits of the ID is transmitted first. If the value of an 11-bit standard ID is equal to or smaller than the first 11 bits of a 29-bit extended ID, the 11-bit standard ID has a higher priority than a message frame with a 29-bit extended ID.
2	Frame type	A data frame with an 11-bit standard ID (FCNnMmSTRB.FCNnMmSSRT cleared to 0) has a higher priority than a remote frame with a standard ID and a message frame with an extended ID.
3	ID type	A message frame with a standard ID (message buffer identifier register FCNnMmMID... bit FCNnMmSSIE is cleared to 0) has a higher priority than a message frame with an extended ID.
4	Value of lower 18 bits of ID [ID17 to ID0]:	If one or more transmission-pending extended ID message frame has equal values in the first 11 bits of the ID and the same frame type (equal FCNnMmSTRB.FCNnMmSSRT bit values), the message frame with the lowest value in the lower 18 bits of its extended ID is transmitted first.
5 (low)	Message buffer number	If two or more message buffers request transmission of message frames with the same ID, the message from the message buffer with the lowest message buffer number is transmitted first.

Notes 1. If the automatic block transmission request bit FCNnGMABCTL.FCNnGMABABTT is set to 1 in the normal operation mode with ABT, FCNnMmCTL.FCNnMmTRQF is set to 1 only for one message buffer in the ABT message buffer group.

If ABT mode is triggered by setting FCNnGMABCTL.FCNnGMABSEAT = 1, then one of the FCNnMmCTL.FCNnMmTRQF in the ABT area (64 message buffers FCN:0 to 15, and 128 message buffers FCN:0 to 31) will be set to 1. Beyond this transmit request, the application can request transmissions (set FCNnMmTRQF to 1) for other TX-message buffers that do not belong to the ABT area. In that case an interval arbitration process (TX-search) evaluates all TX-message buffers with FCNnMmTRQF set to 1 and chooses the message buffer that contains the highest prioritized identifier for the next transmission. If there are 2 or more identifiers that have the highest priority (i.e. identical identifiers), the message located at the lowest message buffer number is transmitted at first.

Upon successful transmission of a message frame, the following operations are performed.

- The FCNnMmCTL.FCNnMmTRQF flag of the corresponding transmit message buffer is automatically cleared to 0.
- The transmission completion status bit FCNnCMISCTL.FCNnCMISITSF0 is set to 1 (if the interrupt enable bit FCNnMmIENF of the corresponding transmit message buffer is set to 1).
- An interrupt request signal INTCnTRX is output (if FCNnCMIECTL.FCNnCMIESEIE0 is set to 1 and if the interrupt enable bit FCNnMmIENF of the corresponding transmit message buffer is set to 1).

2. When changing the contents of a transmit buffer, the FCNnMmCTL.FCNnMmRDYF flag of this buffer must be cleared before updating the buffer contents. As during internal transfer actions, the FCNnMmRDYF flag may be locked temporarily, the status of FCNnMmRDYF must be checked by software, after changing it.

22.9.2 Transmit history list function

The transmit history list (THL) function records in the transmit history list the number of the transmit message buffer from which data or remote frames have been were sent. The THL consists of storage elements equivalent to up to 15 messages (on 64 message buffer FCN) or up to 31 messages (on 128 message buffer FCN), the last out-message pointer FCNnCMLOSTR[7:0] with the corresponding FCNnCMLOSTR register, and the transmit history list get pointer FCNnCMTGSSPT[7:0] with the corresponding FCNnCMTGTX register.

The THL is undefined immediately after the transition of the FCN module from the initialization mode to one of the operation modes.

The FCNnCMLOSTR register holds the contents of the THL element indicated by the value of the FCNnCMLOSTR.FCNnCMLOSTR[7:0] pointer minus 1. By reading the FCNnCMLOSTR register, therefore, the number of the message buffer that transmitted a data frame or remote frame first can be checked. The FCNnCMLOSTR[7:0] pointer is utilized as a write pointer that indicates to what part of the THL a message buffer number is recorded. Any time a data frame or remote frame is transmitted, the corresponding message buffer number is recorded to the THL element indicated by the FCNnCMLOSTR[7:0] pointer. Each time recording to the THL has been completed, the FCNnCMLOSTR[7:0] pointer is automatically incremented. In this way, the number of the message buffer that has received and stored a frame will be recorded chronologically.

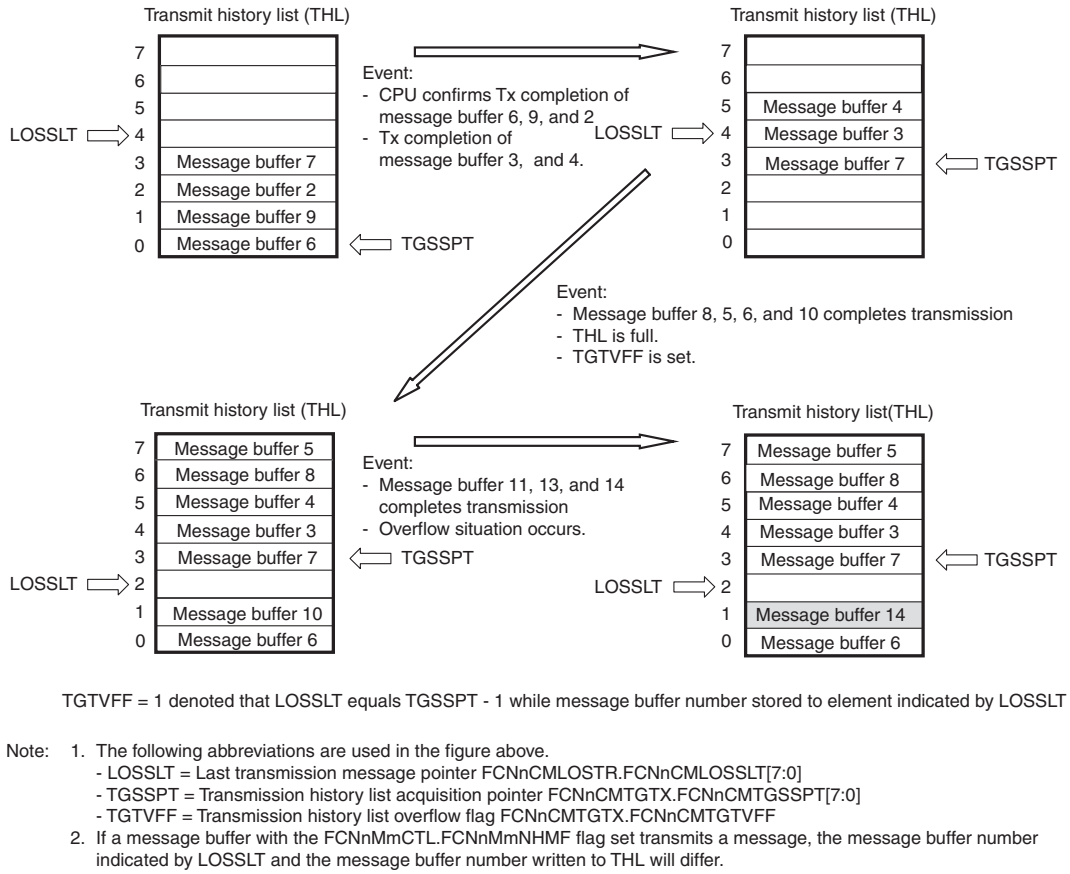
For message buffers, where the flag FCNnMmCTL.FCNnMmNHMF is set, no entry in the history lists is recorded.

The FCNnCMTGTX.FCNnCMTGSSPT[7:0] pointer is utilized as a read pointer that reads a recorded message buffer number from the THL. This pointer indicates the first THL element that the CPU has not yet read. By reading the FCNnCMTGTX register by software, the number of a message buffer that has completed transmission can be read. Each time a message buffer number is read from the FCNnCMTGTX register, the FCNnCMTGSSPT[7:0] pointer is automatically incremented.

If the value of the FCNnCMTGTX.FCNnCMTGSSPT[7:0] pointer matches the value of the FCNnCMLOSTR.FCNnCMLOSTR[7:0] pointer, FCNnCMTGTX.FCNnCMTGSSPM (transmit history list pointer match) is set to 1. This indicates that no message buffer numbers that have not been read remain in the THL. If a new message buffer number is recorded, the FCNnCMLOSTR[7:0] pointer is incremented and because its value no longer matches the value of the FCNnCMTGSSPT[7:0] pointer, FCNnCMTGSSPM is cleared. In other words, the numbers of the unread message buffers exist in the THL.

If the FCNnCMLOSTR.FCNnCMLOSTR[7:0] pointer is incremented and matches the value of the FCNnCMTGTX.FCNnCMTGSSPT[7:0] pointer minus 1, FCNnCMTGTX.FCNnCMTGTVFF (transmit history list overflow) is set to 1. This indicates that the THL is full of message buffer numbers that have not been read. If a new message is received and stored, the message buffer number recorded last is overwritten by the message buffer number that transmitted its message afterwards. In this case, after FCNnCMTGTVFF has been set (1), therefore, the recorded message buffer numbers in the THL do not completely reflect the chronological order. Even in this case, however, the CPU can identify the number of the message buffer that completed reception by searching all reception buffers (the CPU does this before resetting transmission). Regardless of the FCNnCMTGTX.FCNnCMTVFF setting, 14 (64 message buffers) or 30 (128 message buffer) transmit message buffer numbers are stored in THL.

Caution If the history list is in the overflow condition (FCNnCMTGTX.FCNnCMTGTVFF is set), reading the history list contents is still possible, until the history list is empty (indicated by FCNnCMTGTX.FCNnCMTGSSPM flag set). Nevertheless, the history list remains in the overflow condition, until FCNnCMTGTVFF is cleared by software. If FCNnCMTGTVFF is not cleared, the FCNnCMTGTX.FCNnCMTGSSPM flag will also not be updated (cleared) upon successful transmission of a new message. This may lead to the situation, that FCNnCMTGSSPM indicates an empty history list, although a successful transmission has taken place, while the history list is in the overflow state (FCNnCMTGTVFF and FCNnCMTGSSPM are set).



TGTVFF = 1 denoted that LOSSLT equals TGSSPT - 1 while message buffer number stored to element indicated by LOSSLT - 1.

Note: 1. The following abbreviations are used in the figure above.
- LOSSLT = Last transmission message pointer FCNnCMLOSTR.FCNnCMLOSSLT[7:0]
- TGSSPT = Transmission history list acquisition pointer FCNnCMTGTX.FCNnCMTGSSPT[7:0]
- TGTVFF = Transmission history list overflow flag FCNnCMTGTX.FCNnCMTGTVFF

2. If a message buffer with the FCNnMmCTL.FCNnMmNHMF flag set transmits a message, the message buffer number indicated by LOSSLT and the message buffer number written to THL will differ.

Figure 22-10 Transmit history list

22.9.3 Automatic block transmission (ABT)

The automatic block transmission (ABT) function is used to transmit two or more data frames successively with no CPU interaction. The maximum number of transmit message buffers assigned to the ABT function is 16 (for 64 message buffer FCN) or 32 (for 128 message buffer FCN), always located in the lowest message buffers.

By setting FCNnCMCLCTL.FCNnCMCLMDOF[2:0] to 010_B, “normal operation mode with automatic block transmission function” (hereafter referred to as ABT mode) can be selected.

To issue an ABT transmission request, define the message buffers by software first. Set FCNnMmSTRB.FCNnMmSSAM = 1 in all the message buffers used for ABT, and define all the buffers as transmit message buffers by setting the FCNnMmSTRB.FCNnMmSSMT[3:0] bits to 0000_B. Be sure to set the same ID for the message buffers for ABT even when that ID is being used for all the message buffers. To use two or more IDs, set the ID of each message buffer by using the FCNnMmMID0H and FCNnMmMID1H or FCNnMmMID0W registers. Set the FCN message data bytes registers before issuing a transmission request for the ABT function.

After initialization of message buffers for ABT is finished, FCNnMmCTL.FCNnMmRDYF needs to be set to 1. In the ABT mode, FCNnMmCTL.FCNnMmTRQF does not have to be manipulated by software.

After the data for the ABT message buffers has been prepared, set FCNnGMABCTL.FCNnGMABSEAT = 1. Automatic block transmission is then started. When ABT is started, FCNnMmCTL.FCNnMmTRQF in the first message buffer (message buffer 0) is automatically set to 1. After transmission of the data of message buffer 0 is finished, the FCNnMmTRQF of the next message buffer, message buffer 1, is set automatically. In this way, transmission is executed successively.

A delay time can be inserted by program in the interval in which the transmission request FCNnMmCTL.FCNnMmTRQF is automatically set while successive transmission is being executed. The delay time to be inserted is defined by the FCNnGMADCTL register. The unit of the delay time is DBT (data bit time). DBT depends on the setting of the FCNnCMBRPRS and FCNnCMBTCTL registers.

Among transmit objects within the ABT-area, the priority of the transmission ID is not evaluated. Messages are sent by order of message number, starting with message buffer 0. When the transmission of the data frame from the last message buffer is complete, FCNnGMABCTL.FCNnGMABABTT is automatically cleared to 0, and ABT operation completes.

If there is an ABT message buffer for which FCNnMmCTL.FCNnMmRDYF is cleared during ABT, no data frame is transmitted from that buffer, ABT is stopped, and FCNnGMABCTL.FCNnGMABABTT is cleared. After that, transmission can be resumed from the message buffer where ABT stopped, by setting FCNnMmRDYF and FCNnGMABABTT to 1 by software. To not resume transmission from the message buffer where ABT stopped, the internal ABT engine can be reset by setting the FCNnGMABCTL.FCNnGMABCLRF bit to 1 while ABT mode is stopped and FCNnGMABABTT is cleared to 0. In this case, transmission is started from message buffer 0 if FCNnGMABCTL.FCNnGMABSEAC is cleared to 0 and then FFCNnGMABABTT is set to 1.

An interrupt can be used to check if data frames have been transmitted from all the message buffers for ABT. To do so, FCNnMmCTL.FCNnMmIENF of each message buffer except the last message buffer needs to be cleared (0).

If a transmit message buffer other than those used by the ABT function is assigned to a transmit message buffer, the message to be transmitted next is determined by the priority of the transmission ID of the ABT message buffer whose transmission is currently held pending and the transmission ID of the message buffers other than those used by the ABT function.

Transmission of a data frame from an ABT message buffer is not recorded in the transmit history list (THL).

Cautions

1. Set FCNnGMABCTL.FCNnGMABSEAC = 1 while FCNnGMABCTL.FCNnGMABABTT is cleared to 0 in order to resume ABT operation at buffer No. 0. If FCNnGMABSEAC is set to 1 while FCNnGMABABTT is set to 1, the subsequent operation is not guaranteed.
2. If the automatic block transmission engine is cleared by setting FCNnGMABCTL.FCNnGMABSEAC = 1, FCNnGMABSEAC is automatically cleared immediately after the processing of the clearing request is completed.
3. Do not trigger automatic block transmission in the initialization mode. If FCNnGMABCTL.FCNnGMABSEAT is set in the initialization mode, the proper operation is not guaranteed after the mode is changed from the initialization mode to the ABT mode.
4. Do not set FCNnMmCTL.FCNnMmTRQF of the ABT message buffers to 1 by software in the normal operation mode with ABT. Otherwise, the operation is not guaranteed.
5. The FCNnGMADCTL register is used to set the delay time that is inserted in the period from completion of the preceding ABT message to setting of FCNnMmCTL.FCNnMmTRQF for the next ABT message when the transmission requests are set in the order of message numbers for each message for ABT that is successively transmitted in the ABT mode. The timing at which the messages are actually transmitted onto the CAN bus varies depending on the status of transmission from other stations and the status of the setting of the transmission request for messages other than the ABT messages.
6. If a transmission request is made for a message other than an ABT message and if no delay time is inserted in the interval in which transmission requests for ABT are automatically set (FCNnGMADCTL = 00_H), messages other than ABT messages may be transmitted not depending on their priority compared to the priority of the ABT message.
7. Do not clear FCNnMmCTL.FCNnMmRDYF to 0 when FCNnGMABCTL.FCNnGMABABTT = 1.

22.9.4 Transmission abort process

(1) Transmission abort process except for in normal operation mode with automatic block transmission (ABT)

The user can clear FCNnMmCTL.FCNnMmTRQF to 0 to abort a transmission request. FCNnMmTRQF will be cleared immediately if the abort was successful. Whether the transmission was successfully aborted or not can be checked using FCNnCMCLCTL.FCNnCMCLSSTS and the FCNnCMTGTX register, or the FCNnMmCTL.FCNnMmTCPF flag, which indicate the transmission status on the CAN bus (for details, refer to the processing in *Figure 22-25 "Transmission abort processing (except normal operation mode with ABT)" on page 1582*).

(2) Transmission abort process for ABT transmission in normal operation mode with automatic block transmission (ABT)

<R> To abort ABT that is already started, clear FCNnGMABCTL.FCNnGMABABTT to 0. In this case, FCNnGMABCTL.FCNnGMABABTT remains 1 if an ABT message is currently being transmitted and until the transmission is completed (successfully or not), and is cleared to 0 as soon as transmission is finished. This aborts ABT.

If the last transmission (before ABT) was successful, the normal operation mode with ABT is left with the internal ABT pointer pointing to the next message buffer to be transmitted.

In the case of an erroneous transmission, the position of the internal ABT pointer depends on the status of FCNnMmCTL.FCNnMmTRQF in the last transmitted message buffer. If FCNnMmTRQF is cleared to 0 when clearing FCNnGMABCTL.FCNnGMABABTT is requested, the internal ABT pointer is incremented (+1) and points to the next message buffer in the ABT area (for details, refer to the process in *Figure 22-27 "ABT transmission request abort processing (in normal operation mode with ABT) (1)" on page 1584*).

<R> **Caution** Be sure to abort ABT by clearing FCNnGMABCTL.FCNnGMABABTT to 0. The operation is not guaranteed if aborting transmission is requested by clearing FCNnMmCTL.FCNnMmRDYF.

When the normal operation mode with ABT is resumed after ABT has been aborted and FCNnGMABCTL.FCNnGMABSEAT is set to 1, the next ABT message buffer to be transmitted can be determined from the following table.

Status of FCNnMmCTL. FCNnMmTRQF of ABT message buffer	Abort after successful transmission	Abort after erroneous transmission
Set (1)	Next message buffer in the ABT area[a]	Same message buffer in the ABT area
Cleared (0)	Next message buffer in the ABT area[a]	Next message buffer in the ABT area[a]

a) The above resumption operation can be performed only if a message buffer ready for ABT exists in the ABT area. For example, an abort request that is issued while ABT of highest ABT message buffer is in progress is regarded as completion of ABT, rather than abort, if transmission of this message buffer has been successfully completed, even if FCNnGMABCTL.FCNnGMABABTT is cleared to 0. If FCNnMmCTL.FCNnMmRDYF in the next message buffer in the ABT area is cleared to 0, the internal ABT pointer is retained, but the resumption operation is not performed even if FCNnGMABABTT is set to 1, and ABT ends immediately.

22.9.5 Remote frame transmission

Remote frames can be transmitted only from transmit message buffers. Set whether a data frame or remote frame is transmitted via FCNnMmSTRB.FCNnMmSSRT. Setting FCNnMmSSRT = 1 sets remote frame transmission.

22.10 Power Saving Modes

22.10.1 FCN sleep mode

The FCN sleep mode can be used to set the CAN Controller to stand-by mode in order to reduce power consumption. The FCN module can enter the FCN sleep mode from all operation modes. Release of the FCN sleep mode returns the FCN module to exactly the same operation mode from which the FCN sleep mode was entered.

In the FCN sleep mode, the FCN module does not transmit messages, even when transmission requests are issued or pending.

(1) Entering FCN sleep mode

The CPU issues a FCN sleep mode transition request by setting FCNnCMCLCTL.FCNnCMCLMDPF[1:0] = 01_B.

This transition request is acknowledged only under the following conditions.

1. The FCN module is already in one of the following operation modes
   - Normal operation mode
   - Normal operation mode with ABT
   - Receive-only mode
   - Single-shot mode
   - Self-test mode
   - FCN stop mode in all the above operation modes
2. The CAN bus state is bus idle (the 4th bit in the interframe space is recessive).
   If the CAN bus is fixed to dominant, the request for transition to the FCN sleep mode is held pending. Also the transition from FCN stop mode to FCN sleep mode is independent of the CAN bus state.
3. No transmission request is pending.
4. <R> Power save mode cannot be set in combination with the change of operation mode. Be sure to perform these operations in different steps.

Note If a sleep mode request is pending, and at the same time a message is received in a message box, the sleep mode request is not cancelled, but is executed right after message storage has been finished. This may result in FCN being in sleep mode, while the CPU would execute the RX interrupt routine. Therefore, the interrupt routine must check the access to the message buffers as well as reception history list registers by using the FCNnGMCLSSMO flag, if sleep mode is used.

If any one of the conditions mentioned above is not met, the FCN module will operate as follows.

- If the FCN sleep mode is requested from the initialization mode, the FCN sleep mode transition request is ignored and the FCN module remains in the initialization mode.
- If the CAN bus state is not bus idle (i.e., the CAN bus state is either transmitting or receiving) when the FCN sleep mode is requested in one of the operation modes, immediate transition to the FCN sleep mode is not possible. In this case, the FCN sleep mode transition request is held pending until the CAN bus state becomes bus idle (the 4th bit in the interframe space is recessive). In the time from the FCN sleep mode request to successful transition, FCNnCMCLCTL.FCNnCMCLMDPF[1:0]

remain 00_B. When the module has entered the FCN sleep mode, the FCNnCMCLMDPF[1:0] bits are set to 01_B.

- If a request for transition to the initialization mode and a request for transition to the FCN sleep mode are made at the same time while the FCN module is in one of the operation modes, the request for the initialization mode is enabled. The FCN module enters the initialization mode at a predetermined timing. At this time, the FCN sleep mode request is not held pending and is ignored.
- Even when initialization mode and sleep mode are not requested simultaneously (i.e the first request has not been granted while the second request is made), the request for initialization has priority over the sleep mode request. The sleep mode request is cancelled when the initialization mode is requested. When a pending request for initialization mode is present, a subsequent request for Sleep mode request is cancelled right at the point in time where it was submitted.

(2) Status in FCN sleep mode

The FCN module is in the following state after it enters the FCN sleep mode:

- The internal operating clock is stopped and the power consumption is minimized.
- The function to detect the falling edge of the FCN reception pin (CRXDn) remains in effect to wake up the FCN module from the CAN bus.
- To wake up the FCN module from the CPU, data can be set to FCNnCMCLCTL.FCNnCMCLMDPF[1:0], but nothing can be written to other FCN module registers or bits.
- The FCN module registers can be read, except for the FCNnCMLISTR, FCNnCMRGRX, FCNnCMLOSTR, and FCNnCMTGTX registers.
- The FCN message buffer registers cannot be written or read.
- FCNnGMCLCTL.FCNnGMCLSSMO is cleared.
- The registers FCNnDNBMRX cannot be read.
- A request for transition to the initialization mode is not acknowledged and is ignored.

(3) Releasing FCN sleep mode

The FCN sleep mode is released by the following events:

- When the CPU sets FCNnCMCLCTL.FCNnCMCLMDPF[1:0] to 00_B
- A falling edge at the FCN reception pin CRXDn (i.e. the CAN bus level shifts from recessive to dominant)

Caution Even if the falling edge belongs to the SOF of a receive message, this message will not be received and stored. If the CPU has turned off the clock supply to the FCN module while the FCN module was in sleep mode, even subsequently the FCN sleep mode will not be released and FCNnCMCLMDPF[1:0] will remain 01_B unless the clock to the FCN module is supplied again. In addition to this, the receive message will not be received after that.

After releasing the sleep mode, the FCN module returns to the operation mode from which the FCN sleep mode was requested and FCNnCMCLCTL.FCNnCMCLCTL[1:0] must be reset by software to 00_B. If the FCN sleep mode is released by a change in the CAN bus state, FCNnCMISCTL.FCNnCMISITSF5 is set to 1, regardless of FCNnCMIECTL.FCNnCMIEINTF[6:0]. After the FCN module is released from the FCN sleep mode, it participates in the CAN bus again by automatically detecting 11 consecutive recessive-level bits on the CAN bus. The user application has to wait until FCNnGMCLCTL.FCNnGMCLSSMO = 1, before accessing message buffers again.

When a request for transition to the initialization mode is made while the FCN module is in the FCN sleep mode, that request is ignored; the FCN module has to be released from sleep mode by software first before entering the initialization mode.

Cautions

1. Be aware that the release of FCN sleep mode by CAN bus event, and thus the wake up interrupt may happen at any time, even right after requesting sleep mode, if a CAN bus event occurs.
2. Always reset the FCNnCMCLCTL.FCNnCMCLMDPF[1:0] bits to 00_B, when waking up from FCN sleep mode, before accessing any other registers of the FCN module.
3. Always clear the interrupt flag FCNnCMISCTL.FCNnCMISITSF5, when waking up from FCN sleep mode.

22.10.2 FCN stop mode

The FCN stop mode can be used to set the CAN controller to stand-by mode to reduce power consumption. The FCN module can enter the FCN stop mode only from the FCN sleep mode. Release of the FCN stop mode puts the FCN module in the FCN sleep mode.

The FCN stop mode can only be released (entering FCN sleep mode) by setting 01_B to FCNnCMCLCTL.FCNnCMCLMDPF[1:0] and not by a change in the CAN bus state. No message is transmitted even when transmission requests are issued or pending.

(1) Entering FCN stop mode

A FCN stop mode transition request is issued by setting 11_B to FCNnCMCLCTL.FCNnCMCLMDPF[1:0].

A FCN stop mode request is only acknowledged when the FCN module is in the FCN sleep mode. In all other modes, the request is ignored.

Caution To set the FCN module to the FCN stop mode, the module must be in the FCN sleep mode. To confirm that the module is in the sleep mode, check that the FCNnCMCLCTL.FCNnCMCLMDPF[1:0] = 01_B, and then request the FCN stop mode. If a bus change occurs at the FCN reception pin CRXDn while this process is being performed, the FCN sleep mode is automatically released. In this case, the FCN stop mode transition request cannot be acknowledged.

(2) Status in FCN stop mode

The FCN module is in the following state after it enters the FCN stop mode.

- The internal operating clock is stopped and the power consumption is minimized.
- To wake up the FCN module from the CPU, data can be set to FCNnCMCLCTL.FCNnCMCLMDPF[1:0], but nothing can be written to other FCN module registers or bits.
- The FCN module registers can be read, except for the FCNnCMLISTR, FCNnCMRGRX, FCNnCMLOSTR, and FCNnCMTGTX registers.
- The FCN message buffer registers cannot be written or read.
- FCNnGMCLCTL.FCNnGMCLSSMO is cleared.
- The registers FCNnDNBMRX cannot be read.
- An initialization mode transition request is not acknowledged and is ignored.

(3) Releasing FCN stop mode

The FCN stop mode can only be released by writing 01_B to FCNnCMCLCTL.FCNnCMCLMDPF[1:0]. After releasing the FCN stop mode, the FCN module enters the FCN sleep mode.

When the initialization mode is requested while the FCN module is in the FCN stop mode, that request is ignored; the CPU has to release the stop mode and subsequently FCN sleep mode before entering the initialization mode. It is impossible to enter the other operation mode directly from the FCN stop mode not entering the FCN sleep mode, that request is ignored.

22.10.3 Example of using power saving modes

In some application systems, it may be necessary to place the CPU in a power saving mode to reduce the power consumption. By using the power saving mode specific to the FCN module and the power saving mode specific to the CPU in combination, the CPU can be woken up from the power saving status by the CAN bus.

Here is an example for using the power saving modes.

- First, put the FCN module in the FCN sleep mode (FCNnCMCLCTL.FCNnCMCLMDPF[1:0] = 01_B).
  After successfully confirming this state by reading back the sleep mode status, put the CPU in the power saving mode. Disable interrupts for the CPU, while processing additional tasks after the FCN module is in sleep mode, to avoid that the FCN wakeup interrupt is acknowledged.
  If a rising edge from recessive to dominant is detected on the CRXDn FCN reception pin in this state, FCNnCMISCTL.FCNnCMISITSF5 in the FCN module will be set to 1. If FCNnCMIECTL.FCNnCMIEINT5 is set to 1, a wakeup interrupt (INTCnWUP) is generated.
  The FCN module is automatically released from FCN sleep mode (FCNnCMCLMDPF[1:0] = 00_B) and returns to normal operation mode.
- The CPU, in response to INTCnWUP, can release its own power saving mode and return to normal operation mode.

  To further reduce the power consumption of the CPU, the internal clock - including that of the FCN module - may be stopped. In this case, the operating clock supplied to the FCN module is stopped after the FCN module has been put in FCN sleep mode. Then the CPU enters a power saving mode in which the clock supplied to the CPU is stopped.
- If an edge transition from recessive to dominant is detected at the FCN reception pin CRXDn in this status, the FCN module can set FCNnCMISCTL.FCNnCMISITSF5 to 1 and generate the wakeup interrupt INTCnWUP even if it is not supplied with the clock.
- The other functions, however, do not operate, because clock supply to the FCN module is stopped, and the module remains in FCN sleep mode.
- The CPU, in response to INTCnWUP,
  - releases its power saving mode,
  - resumes supply of the internal clocks - including the clock to the FCN module - after the oscillation stabilization time has elapsed, and
  - starts instruction execution.
- The FCN module is immediately released from the FCN sleep mode when clock supply is resumed, and returns to the normal operation mode (FCNnCMCLCTL.FCNnCMCLMDPF[1:0] = 00_B).

22.11 Interrupt Function

The FCN module provides 6 different interrupt sources.

The occurrence of these interrupt sources is stored in interrupt status registers. Four separate interrupt request signals are generated from the six interrupt sources. When an interrupt request signal that corresponds to two or more interrupt sources is generated, the interrupt sources can be identified by using an interrupt status register. After an interrupt source has occurred, the corresponding interrupt status bit must be cleared to 0 by software.

Table 22-20 List of FCN module interrupt sources

No.	Interrupt status bit FCNnCMISCTL.	Interrupt enable bit FCNnCMIECTL. [a]	Interrupt request signal	Interrupt source description
1	FCNnCMISITSF0	FCNnCMIESEIE0	INTCnTRX	Message frame successfully transmitted from message buffer m
2	FCNnCMISITSF1	FCNnCMIESEIE1	INTCnREC	Valid message frame reception in message buffer m
3	FCNnCMISITSF2	FCNnCMIESEIE2	INTCnERR	FCN module error state interrupt • This interrupt is generated when the transmission/reception error counter is at the warning level, or in the error passive or bus-off state.
4	FCNnCMISITSF3	FCNnCMIESEIE3		FCN module protocol error interrupt • This interrupt is generated when a stuff error, form error, ACK error, bit error, or CRC error occurs.
5	FCNnCMISITSF4	FCNnCMIESEIE4		FCN module arbitration loss interrupt
6	FCNnCMISITSF5	FCNnCMIESEIE5	INTCnWUP	FCN module wakeup interrupt from FCN sleep mode • This interrupt is generated when the FCN module wakes up from FCN sleep mode, due to detection of a rise on the FCN reception pin (change of CAN bus from recessive to dominant).
7	FCNnCMISITSF6	FCNnCMIESEIE6		FCN module transmit abort interrupt status • This interrupt is generated when the abortion of a transmission was successful (aborted message was not sent).

a) The message buffer interrupt enable bit FCNnMmCTL.FCNnMmIENF of the corresponding message buffer has to be set to 1 for that message buffer to participate in the interrupt generation process.

22.12 Diagnosis Functions and Special Operational Modes

The FCN module provides a receive-only mode, single-shot mode, and self-test mode to support CAN bus diagnosis functions or the operation of special CAN communication methods.

22.12.1 Receive-only mode

The receive-only mode is used to monitor receive messages without causing any interference on the CAN bus and can be used for CAN bus analysis nodes.

For example, this mode can be used for automatic baud-rate detection. The baud rate in the FCN module is changed until “valid reception” is detected, so that the baud rates in the module match (“valid reception” means a message frame has been received in the CAN protocol layer without occurrence of an error and with an appropriate ACK between nodes connected to the CAN bus). A valid reception does not require message frames to be stored in a receive message buffer (data frames) or transmit message buffer (remote frames). Whether a reception is valid can be checked by confirming that FCNnCMCLCTL.FCNnCMCLVALF is set to 1.

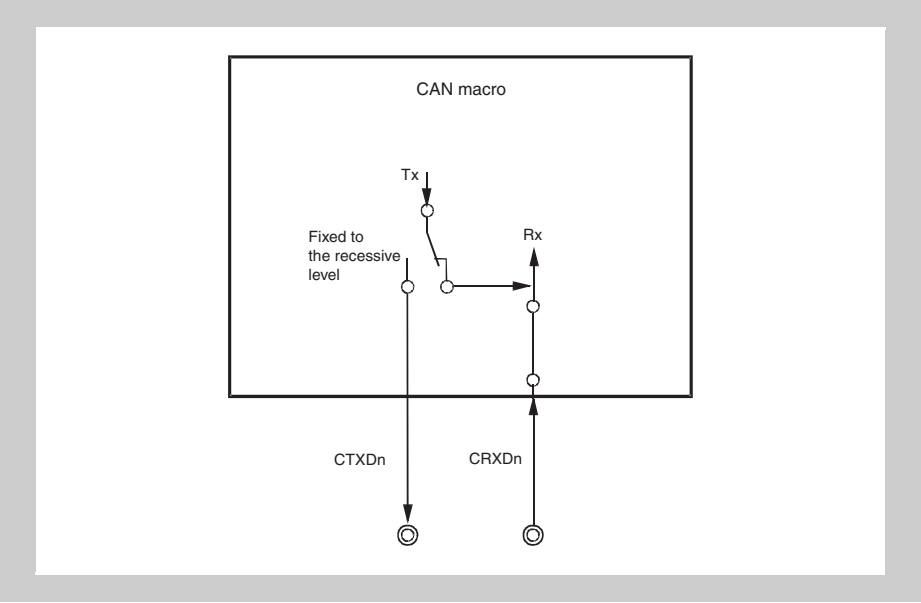


Figure 22-11 FCN module terminal connection in receive-only mode

In the receive-only mode, no message frames can be transmitted from the FCN module to the CAN bus. Transmit requests issued for message buffers defined as transmit message buffers are held pending.

In the receive-only mode, the FCN transmission pin CTXDn in the FCN module is fixed to the recessive level. Therefore, no active error flag can be transmitted from the FCN module to the CAN bus even when a CAN bus error is detected while receiving a message frame. Since no transmission can be issued from the FCN module, the transmission error counter the FCNnCMERCNT.TEC7 to FCNnCMERCNT.TEC0 bits are never updated. Therefore, a FCN module in the receive-only mode does not enter the bus-off state.

Furthermore, in the receive-only mode ACK is not returned to the CAN bus in this mode upon the valid reception of a message frame. Internally, the local node recognizes that it has transmitted ACK. An overload frame cannot be transmitted to the CAN bus.

Caution If only two CAN nodes are connected to the CAN bus and one of them is operating in the receive-only mode, there is no ACK on the CAN bus. Due to the missing ACK, the transmitting node will transmit an active error flag, and repeat transmitting a message frame. The transmitting node becomes error passive after transmitting the message frame 16 times (assuming that the error counter was 0 in the beginning and no other errors have occurred). After the message frame for the 17th time is transmitted, the transmitting node generates a passive error flag. The receiving node in the receive-only mode detects the first valid message frame at this point, and the FCNnCMCLCTL.FCNnCMCLVALF bit is set to 1 for the first time.

22.12.2 Single-shot mode

In the single-shot mode, automatic re-transmission as defined in the CAN protocol is switched off. (According to the CAN protocol, a message frame transmission that has been aborted by either arbitration loss or error occurrence has to be repeated without control by software.) All other behavior of single shot mode is identical to normal operation mode. Features of single shot mode can not be used in combination with normal mode with ABT.

The single-shot mode disables the re-transmission of an aborted message frame transmission according to the setting of FCNnCMCLCTL.FCNnCMCLALBF. When FCNnCMCLALBF is cleared to 0, re-transmission upon arbitration loss and upon error occurrence is disabled. If FCNnCMCLALBF is set to 1, re-transmission upon error occurrence is disabled, but re-transmission upon arbitration loss is enabled. As a consequence, FCNnMmCTL.FCNnMmTRQF in a message buffer defined as a transmit message buffer is cleared to 0 by the following events:

- Successful transmission of the message frame
- Arbitration loss while sending the message frame
- Error occurrence while sending the message frame

The events arbitration loss and error occurrence can be distinguished by checking FCNnCMISCTL.FCNnCMISITSF4 and FCNnCMISCTL.FCNnCMISITSF3 respectively, and the type of the error can be identified by reading FCNnCMLCSTR.FCN0CMLCSSLC[2:0].

Upon successful transmission of the message frame, the transmit completion interrupt bit FCNnCMISCTL.FCNnCMISITSF0 is set to 1. If FCNnCMIECTL.FCNnCMIEINTF0 is set to 1 at this time, an interrupt request signal is output.

The single-shot mode can be used when emulating time-triggered communication methods (e.g., TTCAN level 1).

Caution FCNnCMCLCTL.FCNnCMCLALBF is only valid in single-shot mode. It does not influence the operation of re-transmission upon arbitration loss in the other operation modes.

22.12.3 Self-test mode

In the self-test mode, message frame transmission and message frame reception can be tested without connecting the CAN node to the CAN bus or without affecting the CAN bus.

In the self-test mode, the FCN module is completely disconnected from the CAN bus, but transmission and reception are internally looped back. The FCN transmission pin CTXDn is fixed to the recessive level.

If the falling edge on the FCN reception pin CRXDn is detected after the FCN module has entered the FCN sleep mode from the self-test mode, however, the module is released from the FCN sleep mode in the same manner as the other operation modes. Use the CRXDn FCN reception pin as a port pin in order to keep the module in FCN sleep mode.

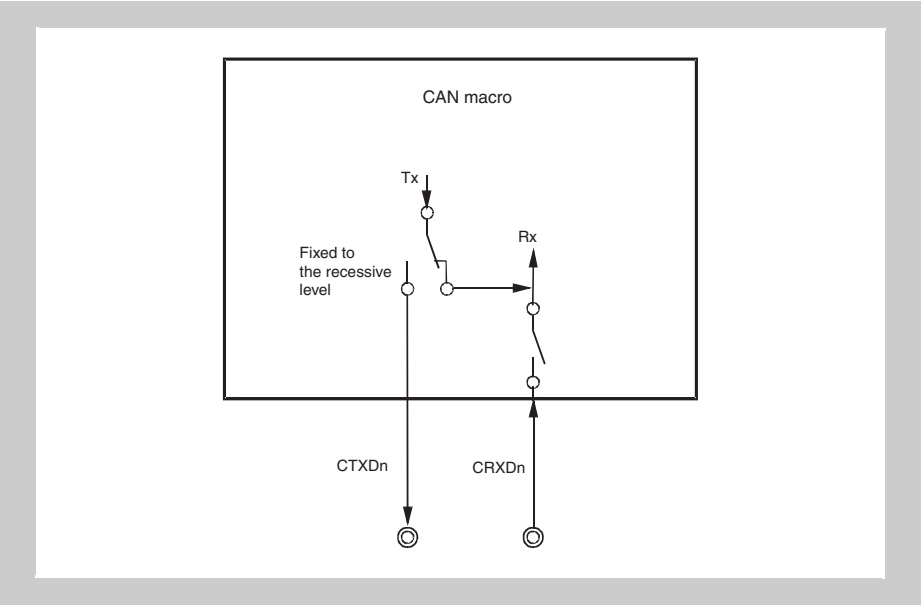


Figure 22-12 FCN module terminal connection in self-test mode

22.12.4 Receive/transmit operation in each operation mode

The following table shows outline of the receive/transmit operation in each operation mode.

Table 22-21 Outline of the receive/transmit in each operation mode

Operation mode	Transmission of data/ remote frame	Transmission of ACK	Transmission of error/ overload frame	Transmission retry	Automatic block transmission (ABT)	Set of FCNnCMCLVALF bit	Store data to message buffer
Initialization mode	No	No	No	No	No	No	No
Normal operation mode	Yes	Yes	Yes	Yes	No	Yes	Yes
Normal operation mode with ABT	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Receive only mode	No	No	No	No	No	Yes	Yes
Single-shot mode	Yes	Yes	Yes	No[a]	No	Yes	Yes
Self-test mode	Yes[b]	Yes[b]	Yes[b]	Yes[b]	No	Yes[b]	Yes[b]

a) When the arbitration lost occurs, control of re-transmission is possible by FCNnCMCLCTL.FCNnCMCLALBF.

b) Each signals are not generated to outside, but generated into the FCN module.

22.13 Time Stamp Function

CAN is an asynchronous serial communication protocol. All nodes connected to the CAN bus have a local, autonomous clock. As a consequence, the clocks of the nodes have no relation (i.e., the clocks are asynchronous and may have different frequencies).

In some applications, however, a common time base over the network (= global time base) is needed. In order to build up a global time base, a time stamp function is used. The essential mechanism of a time stamp function is the capture of timer values triggered by signals on the CAN bus.

22.13.1 Time stamp function

The CAN Controller supports the capturing of timer values triggered by a specific frame. An on-chip 16-bit capture timer unit in a microcontroller system is used in addition to the CAN Controller. The 16-bit capture timer unit captures the timer value according to a trigger signal (TSOUT) for capturing that is output when a data frame is received from the CAN Controller. The CPU can retrieve the time of occurrence of the capture event, i.e., the time stamp of the message received from the CAN bus, by reading the captured value. The TSOUT signal can be selected from the following two event sources and is specified by FCNnCMTSCTL.FCNnCMTSSELE.

- SOF event (start of frame)
  (FCNnCMTSCTL.FCNnCMTSSELE = 0)
- EOF event (last bit of end of frame)
  (FCNnCMTSCTL.FCNnCMTSSELE = 1)

The TSOUT signal is enabled by setting FCNnCMTSCTL.FCNnCMTSTSGE = 1.

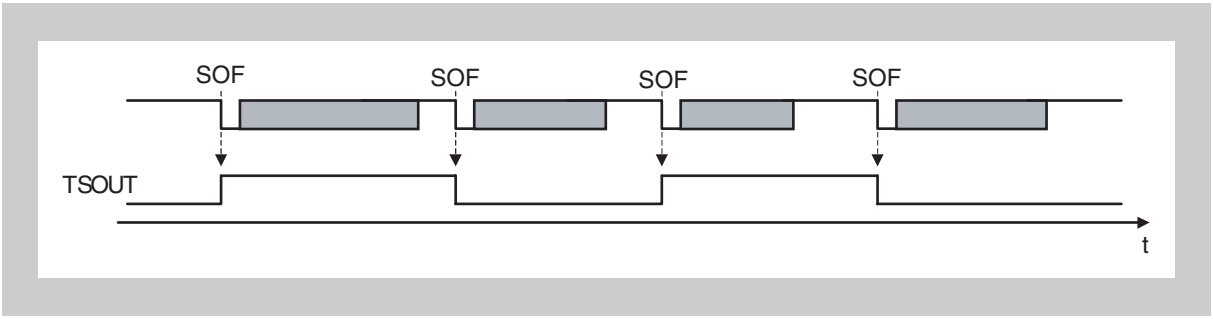


Figure 22-13 Timing diagram of capture signal TSOUT

The TSOUT signal toggles its level upon occurrence of the selected event during data frame reception (in *Figure 22-13 "Timing diagram of capture signal TSOUT"*, the SOF is used as the trigger event source). To capture a timer value by using the TSOUT signal, the capture timer unit must detect the capture signal at both the rising edge and falling edge.

This time stamp function is controlled by the FCNnCMTSLOKE bit of the FCNnCMTSCTL register. When FCNnCMTSLOKE is cleared to 0, the TSOUT signal toggles upon occurrence of the selected event. If FCNnCMTSLOKE is set to 1, the TSOUT signal toggles upon occurrence of the selected event, but the toggle is stopped as FCNnCMTSCTL.FCNnCMTSTSGE is automatically cleared to 0 as soon as the message storing to the message buffer 0 starts. This suppresses the subsequent toggle occurrence by the TSOUT signal, so that the time stamp value toggled last (= captured last) can be saved as the time stamp value of the time at which the data frame was received in message buffer 0.

Caution The time stamp function using the FCNnCMTSLOKE bit stops toggle of the TSOUT signal by receiving a data frame in message buffer 0. Toggle of the TSOUT signal does not stop when a data frame is received in a message buffer other than message buffer 0.
A data frame cannot be received in message buffer 0 when the FCN module is in the normal operation mode with ABT, because message buffer 0 must be set as a transmit message buffer.
In this operation mode, therefore, the function to stop toggle of the TSOUT signal by the FCNnCMTSLOKE bit cannot be used.

22.14 Baud Rate Settings

<R> 22.14.1 Baud rate setting conditions

Make sure that the settings are within the range of limit values for ensuring correct operation of the CAN Controller, as follows.

- 5 TQ ≤ SPT (sampling point) ≤ 17 TQ

  SPT = TSEG1 + 1
- 8 TQ ≤ DBT (data bit time) ≤ 25 TQ

  DBT = TSEG1 + TSEG2 + 1 TQ = TSEG2 + SPT
- 1 TQ ≤ SJW (synchronization jump width) ≤ 4 TQ

  SJW ≤ DBT – SPT
- 4 ≤ TSEG1 ≤ 16
- 1 ≤ TSEG2 ≤ 8

Notes

1. TQ = 1/f_{TQ} (f_{TQ}: CAN protocol layer basic system clock)
2. <R> The values of TSEG1, TSEG2, and SJW are defined by the bits of the FCNnCMBTCTL register as follows:
   TSEG1: FCNnCMBTCTL.FCNnCMBTS1LG[3:0] + 1
   TSEG2: FCNnCMBTCTL.FCNnCMBTS2LG[2:0] + 1
   SJW: FCNnCMBTCTL.FCNnCMBTJWLG[1:0] + 1

Table 22-22 "Settable bit rate combinations" shows the combinations of bit rates that satisfy the above conditions.

Table 22-22 Settable bit rate combinations (1/3)

Valid bit rate setting					FCNnCMBTCTL register setting value		Sampling point (unit: %)
DBT length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
25	1	8	8	8	1111	111	68.0
24	1	7	8	8	1110	111	66.7
24	1	9	7	7	1111	110	70.8
23	1	6	8	8	1101	111	65.2
23	1	8	7	7	1110	110	69.6
23	1	10	6	6	1111	101	73.9
22	1	5	8	8	1100	111	63.6
22	1	7	7	7	1101	110	68.2
22	1	9	6	6	1110	101	72.7
22	1	11	5	5	1111	100	77.3
21	1	4	8	8	1011	111	61.9
21	1	6	7	7	1100	110	66.7
21	1	8	6	6	1101	101	71.4
21	1	10	5	5	1110	100	76.2
21	1	12	4	4	1111	011	81.0
20	1	3	8	8	1010	111	60.0

Table 22-22 Settable bit rate combinations (2/3)

Valid bit rate setting					FCNnCMBTCTL register setting value		Sampling point (unit: %)
DBT length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
20	1	5	7	7	1011	110	65.0
20	1	7	6	6	1100	101	70.0
20	1	9	5	5	1101	100	75.0
20	1	11	4	4	1110	011	80.0
20	1	13	3	3	1111	010	85.0
19	1	2	8	8	1001	111	57.9
19	1	4	7	7	1010	110	63.2
19	1	6	6	6	1011	101	68.4
19	1	8	5	5	1100	100	73.7
19	1	10	4	4	1101	011	78.9
19	1	12	3	3	1110	010	84.2
19	1	14	2	2	1111	001	89.5
18	1	1	8	8	1000	111	55.6
18	1	3	7	7	1001	110	61.1
18	1	5	6	6	1010	101	66.7
18	1	7	5	5	1011	100	72.2
18	1	9	4	4	1100	011	77.8
18	1	11	3	3	1101	010	83.3
18	1	13	2	2	1110	001	88.9
18	1	15	1	1	1111	000	94.4
17	1	2	7	7	1000	110	58.8
17	1	4	6	6	1001	101	64.7
17	1	6	5	5	1010	100	70.6
17	1	8	4	4	1011	011	76.5
17	1	10	3	3	1100	010	82.4
17	1	12	2	2	1101	001	88.2
17	1	14	1	1	1110	000	94.1
16	1	1	7	7	0111	110	56.3
16	1	3	6	6	1000	101	62.5
16	1	5	5	5	1001	100	68.8
16	1	7	4	4	1010	011	75.0
16	1	9	3	3	1011	010	81.3
16	1	11	2	2	1100	001	87.5
16	1	13	1	1	1101	000	93.8
15	1	2	6	6	0111	101	60.0
15	1	4	5	5	1000	100	66.7
15	1	6	4	4	1001	011	73.3
15	1	8	3	3	1010	010	80.0
15	1	10	2	2	1011	001	86.7

Table 22-22 Settable bit rate combinations (3/3)

Valid bit rate setting					FCNnCMBTCTL register setting value		Sampling point (unit: %)
DBT length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
15	1	12	1	1	1100	000	93.3
14	1	1	6	6	0110	101	57.1
14	1	3	5	5	0111	100	64.3
14	1	5	4	4	1000	011	71.4
14	1	7	3	3	1001	010	78.6
14	1	9	2	2	1010	001	85.7
14	1	11	1	1	1011	000	92.9
13	1	2	5	5	0110	100	61.5
13	1	4	4	4	0111	011	69.2
13	1	6	3	3	1000	010	76.9
13	1	8	2	2	1001	001	84.6
13	1	10	1	1	1010	000	92.3
12	1	1	5	5	0101	100	58.3
12	1	3	4	4	0110	011	66.7
12	1	5	3	3	0111	010	75.0
12	1	7	2	2	1000	001	83.3
12	1	9	1	1	1001	000	91.7
11	1	2	4	4	0101	011	63.6
11	1	4	3	3	0110	010	72.7
11	1	6	2	2	0111	001	81.8
11	1	8	1	1	1000	000	90.9
10	1	1	4	4	0100	011	60.0
10	1	3	3	3	0101	010	70.0
10	1	5	2	2	0110	001	80.0
10	1	7	1	1	0111	000	90.0
9	1	2	3	3	0100	010	66.7
9	1	4	2	2	0101	001	77.8
9	1	6	1	1	0110	000	88.9
8	1	1	3	3	0011	010	62.5
8	1	3	2	2	0100	001	75.0
8	1	5	1	1	0101	000	87.5
7[a]	1	2	2	2	0011	001	71.4
7[a]	1	4	1	1	0100	000	85.7
6[a]	1	1	2	2	0010	001	66.7
6[a]	1	3	1	1	0011	000	83.3
5[a]	1	2	1	1	0010	000	80.0
4[a]	1	1	1	1	0001	000	75.0

a) Setting with a DBT value of 7 or less is valid only when the value of the FCNnCMBRPRS register is other than 00_H.

Caution The values in *Table 22-22 "Settable bit rate combinations"* do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

22.14.2 Representative examples of baud rate settings

<R> *Table 22-23 "Representative examples of baud rate settings (f_{CANPRE} = 8 MHz)"* and *Table 22-24 "Representative examples of baud rate settings (f_{CANPRE} = 16 MHz)"* show representative examples of baud rate settings.

Table 22-23 Representative examples of baud rate settings (f_{CANPRE} = 8 MHz) (1/2)

<R>

Set baud rate value (unit: kbps)	Division ratio of FCNnCMBRPRS register	FCNnCMBRPRS register set value	Valid bit rate setting (unit: TQ)					FCNnCMBTCTL register setting value		Sampling point (unit: %)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
500	1	00000000	16	1	1	7	7	0111	110	56.3
500	1	00000000	16	1	3	6	6	1000	101	62.5
500	1	00000000	16	1	5	5	5	1001	100	68.8
500	1	00000000	16	1	7	4	4	1010	011	75.0
500	1	00000000	16	1	9	3	3	1011	010	81.3
500	1	00000000	16	1	11	2	2	1100	001	87.5
500	1	00000000	16	1	13	1	1	1101	000	93.8
500	2	00000001	8	1	1	3	3	0011	010	62.5
500	2	00000001	8	1	3	2	2	0100	001	75.0
500	2	00000001	8	1	5	1	1	0101	000	87.5
250	2	00000001	16	1	1	7	7	0111	110	56.3
250	2	00000001	16	1	3	6	6	1000	101	62.5
250	2	00000001	16	1	5	5	5	1001	100	68.8
250	2	00000001	16	1	7	4	4	1010	011	75.0
250	2	00000001	16	1	9	3	3	1011	010	81.3
250	2	00000001	16	1	11	2	2	1100	001	87.5
250	2	00000001	16	1	13	1	1	1101	000	93.8
250	4	00000011	8	1	3	2	2	0100	001	75.0
250	4	00000011	8	1	5	1	1	0101	000	87.5
125	4	00000011	16	1	1	7	7	0111	110	56.3
125	4	00000011	16	1	3	6	6	1000	101	62.5
125	4	00000011	16	1	5	5	5	1001	100	68.8
125	4	00000011	16	1	7	4	4	1010	011	75.0
125	4	00000011	16	1	9	3	3	1011	010	81.3
125	4	00000011	16	1	11	2	2	1100	001	87.5

<R> **Table 22-23 Representative examples of baud rate settings (f_{CANPRE} = 8 MHz) (2/2)**

Set baud rate value (unit: kbps)	Division ratio of FCNnCMBRPRS register	FCNnCMBRPRS register set value	Valid bit rate setting (unit: TQ)					FCNnCMBTCTL register setting value		Sampling point (unit: %)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
125	4	00000011	16	1	13	1	1	1101	000	93.8
125	8	00000111	8	1	3	2	2	0100	001	75.0
125	8	00000111	8	1	5	1	1	0101	000	87.5
100	4	00000011	20	1	7	6	6	1100	101	70.0
100	4	00000011	20	1	9	5	5	1101	100	75.0
100	5	00000100	16	1	7	4	4	1010	011	75.0
100	5	00000100	16	1	9	3	3	1011	010	81.3
100	8	00000111	10	1	3	3	3	0101	010	70.0
100	8	00000111	10	1	5	2	2	0110	001	80.0
100	10	00001001	8	1	3	2	2	0100	001	75.0
100	10	00001001	8	1	5	1	1	0101	000	87.5
83.3	4	00000011	24	1	7	8	8	1110	111	66.7
83.3	4	00000011	24	1	9	7	7	1111	110	70.8
83.3	6	00000101	16	1	5	5	5	1001	100	68.8
83.3	6	00000101	16	1	7	4	4	1010	011	75.0
83.3	6	00000101	16	1	9	3	3	1011	010	81.3
83.3	6	00000101	16	1	11	2	2	1100	001	87.5
83.3	8	00000111	12	1	5	3	3	0111	010	75.0
83.3	8	00000111	12	1	7	2	2	1000	001	83.3
83.3	12	00001011	8	1	3	2	2	0100	001	75.0
83.3	12	00001011	8	1	5	1	1	0101	000	87.5
33.3	10	00001001	24	1	7	8	8	1110	111	66.7
33.3	10	00001001	24	1	9	7	7	1111	110	70.8
33.3	12	00001011	20	1	7	6	6	1100	101	70.0
33.3	12	00001011	20	1	9	5	5	1101	100	75.0
33.3	15	00001110	16	1	7	4	4	1010	011	75.0
33.3	15	00001110	16	1	9	3	3	1011	010	81.3
33.3	16	00001111	15	1	6	4	4	1001	011	73.3
33.3	16	00001111	15	1	8	3	3	1010	010	80.0
33.3	20	00010011	12	1	5	3	3	0111	010	75.0
33.3	20	00010011	12	1	7	2	2	1000	001	83.3
33.3	24	00010111	10	1	3	3	3	0101	010	70.0
33.3	24	00010111	10	1	5	2	2	0110	001	80.0
33.3	30	00011101	8	1	3	2	2	0100	001	75.0
33.3	30	00011101	8	1	5	1	1	0101	000	87.5

Cautions

1. The values in *Table 22-23 "Representative examples of baud rate settings (f_{CANPRE} = 8 MHz)"* do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

<R> 2. Baud rates higher than 500 kbit/s are not allowed with $f_{CANPRE} \leq 8$ MHz.

Table 22-24 Representative examples of baud rate settings (f_{CANPRE} = 16 MHz) (1/2)

<R>

Set baud rate value (unit: kbps)	Division ratio of FCNnCMBRPRS register	FCNnCMBRPRS register set value	Valid bit rate setting (unit: TQ)					FCNnCMBTCTL register setting value		Sampling point (unit: %)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
1000	1	00000000	16	1	1	7	7	0111	110	56.3
1000	1	00000000	16	1	3	6	6	1000	101	62.5
1000	1	00000000	16	1	5	5	5	1001	100	68.8
1000	1	00000000	16	1	7	4	4	1010	011	75.0
1000	1	00000000	16	1	9	3	3	1011	010	81.3
1000	1	00000000	16	1	11	2	2	1100	001	87.5
1000	1	00000000	16	1	13	1	1	1101	000	93.8
1000	2	00000001	8	1	3	2	2	0100	001	75.0
1000	2	00000001	8	1	5	1	1	0101	000	87.5
500	2	00000001	16	1	1	7	7	0111	110	56.3
500	2	00000001	16	1	3	6	6	1000	101	62.5
500	2	00000001	16	1	5	5	5	1001	100	68.8
500	2	00000001	16	1	7	4	4	1010	011	75.0
500	2	00000001	16	1	9	3	3	1011	010	81.3
500	2	00000001	16	1	11	2	2	1100	001	87.5
500	2	00000001	16	1	13	1	1	1101	000	93.8
500	4	00000011	8	1	3	2	2	0100	001	75.0
500	4	00000011	8	1	5	1	1	0101	000	87.5
250	4	00000011	16	1	3	6	6	1000	101	62.5
250	4	00000011	16	1	5	5	5	1001	100	68.8
250	4	00000011	16	1	7	4	4	1010	011	75.0
250	4	00000011	16	1	9	3	3	1011	010	81.3
250	4	00000011	16	1	11	2	2	1100	001	87.5
250	8	00000111	8	1	3	2	2	0100	001	75.0
250	8	00000111	8	1	5	1	1	0101	000	87.5
125	8	00000111	16	1	3	6	6	1000	101	62.5
125	8	00000111	16	1	7	4	4	1010	011	75.0
125	8	00000111	16	1	9	3	3	1011	010	81.3
125	8	00000111	16	1	11	2	2	1100	001	87.5

<R>

Table 22-24 Representative examples of baud rate settings (f_{CANPRE} = 16 MHz) (2/2)

Set baud rate value (unit: kbps)	Division ratio of FCNnCMBRPRS register	FCNnCMBRPRS register set value	Valid bit rate setting (unit: TQ)					FCNnCMBTCTL register setting value		Sampling point (unit: %)
			Length of DBT	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
125	16	00001111	8	1	3	2	2	0100	001	75.0
125	16	00001111	8	1	5	1	1	0101	000	87.5
100	8	00000111	20	1	9	5	5	1101	100	75.0
100	8	00000111	20	1	11	4	4	1110	011	80.0
100	10	00001001	16	1	7	4	4	1010	011	75.0
100	10	00001001	16	1	9	3	3	1011	010	81.3
100	16	00001111	10	1	3	3	3	0101	010	70.0
100	16	00001111	10	1	5	2	2	0110	001	80.0
100	20	00010011	8	1	3	2	2	0100	001	75.0
83.3	8	00000111	24	1	7	8	8	1110	111	66.7
83.3	8	00000111	24	1	9	7	7	1111	110	70.8
83.3	12	00001011	16	1	7	4	4	1010	011	75.0
83.3	12	00001011	16	1	9	3	3	1011	010	81.3
83.3	12	00001011	16	1	11	2	2	1100	001	87.5
83.3	16	00001111	12	1	5	3	3	0111	010	75.0
83.3	16	00001111	12	1	7	2	2	1000	001	83.3
83.3	24	00010111	8	1	3	2	2	0100	001	75.0
83.3	24	00010111	8	1	5	1	1	0101	000	87.5
33.3	30	00011101	24	1	7	8	8	1110	111	66.7
33.3	30	00011101	24	1	9	7	7	1111	110	70.8
33.3	24	00010111	20	1	9	5	5	1101	100	75.0
33.3	24	00010111	20	1	11	4	4	1110	011	80.0
33.3	30	00011101	16	1	7	4	4	1010	011	75.0
33.3	30	00011101	16	1	9	3	3	1011	010	81.3
33.3	32	00011111	15	1	8	3	3	1010	010	80.0
33.3	32	00011111	15	1	10	2	2	1011	001	86.7
33.3	37	00100100	13	1	6	3	3	1000	010	76.9
33.3	37	00100100	13	1	8	2	2	1001	001	84.6
33.3	40	00100111	12	1	5	3	3	0111	010	75.0
33.3	40	00100111	12	1	7	2	2	1000	001	83.3
33.3	48	00101111	10	1	3	3	3	0101	010	70.0
33.3	48	00101111	10	1	5	2	2	0110	001	80.0
33.3	60	00111011	8	1	3	2	2	0100	001	75.0
33.3	60	00111011	8	1	5	1	1	0101	000	87.5

Caution The values in *Table 22-24 "Representative examples of baud rate settings (f_{CANPRE} = 16 MHz)"* do not guarantee the operation of the network system. Thoroughly check the effect on the network system, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

22.15 Operation of the CAN Controller

The processing procedure for showing in this chapter is recommended processing procedure to operate FCN.

Develop the program referring to recommended processing procedure in this chapter.

22.15.1 Initialization

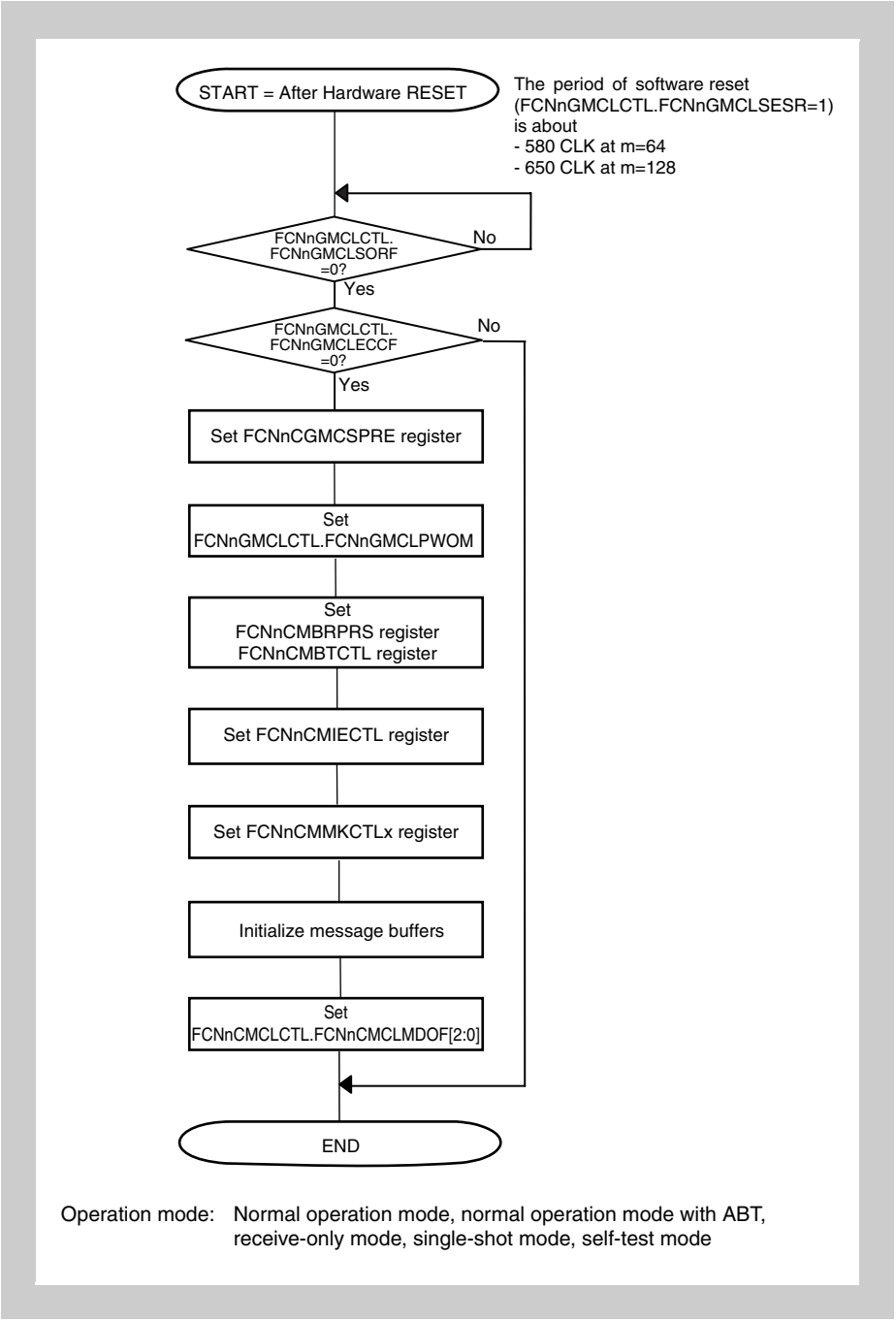


Figure 22-14 Initialization

<R>

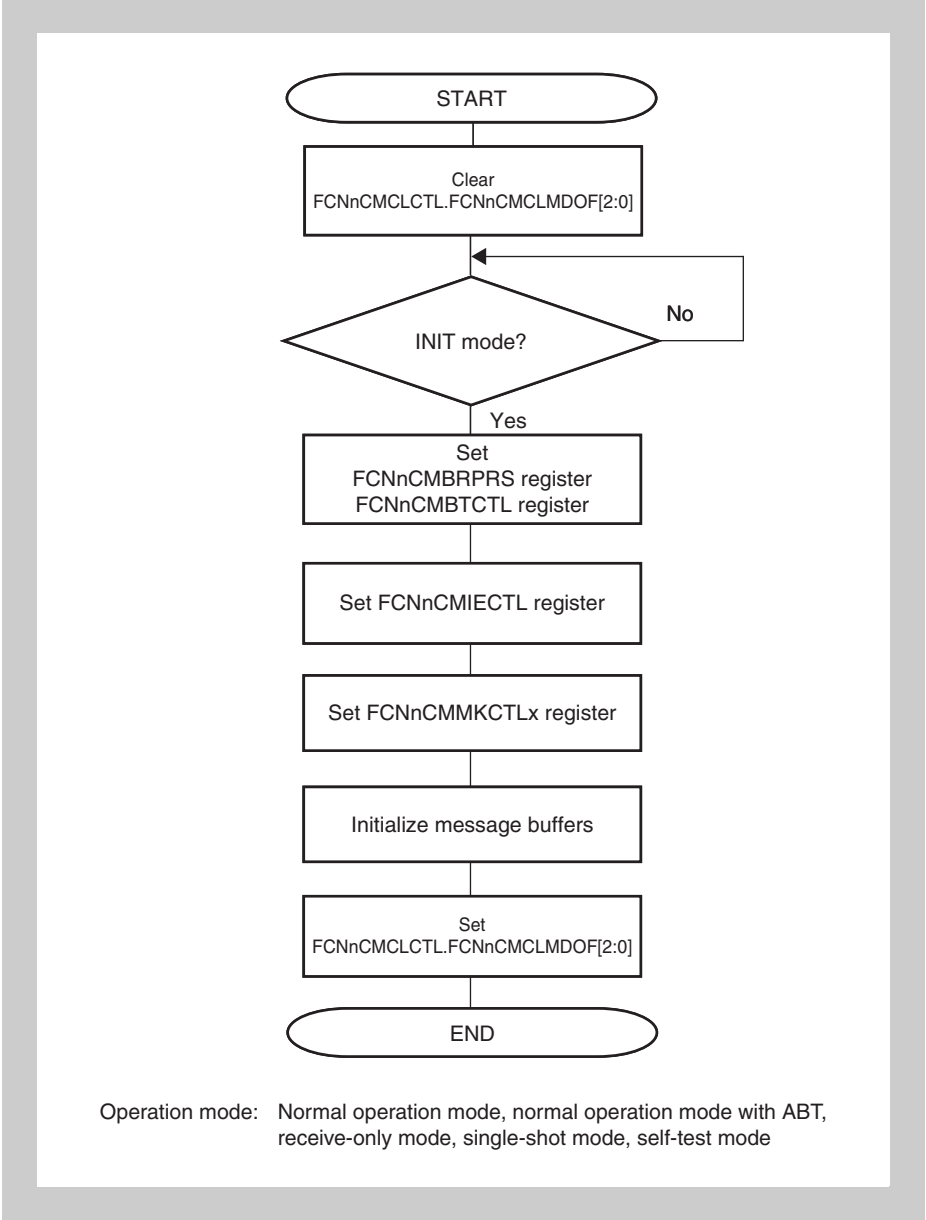


Figure 22-15 Re-initialization

<R> **Caution** Clearing the error counter during re-initialization (by setting FCNnCMCLERCF) must be performed under either of the following conditions:

- In the initialization mode after the FCN module starts up (by changing FCNnGMCLPWOM from 0 to 1)
- In the initialization mode entered after all the transmission requests have been cleared in accordance with the transmission abort processing shown in *Table 22-25 "Transmission abort processing (except normal operation mode with ABT)"* in an operation mode. (In normal operation mode with ABT, clear all the transmission requests in accordance with the transmission abort processing shown in *Table 22-26 "Processing to abort transmission other than ABT transmission (in normal operation mode with ABT)"*.)

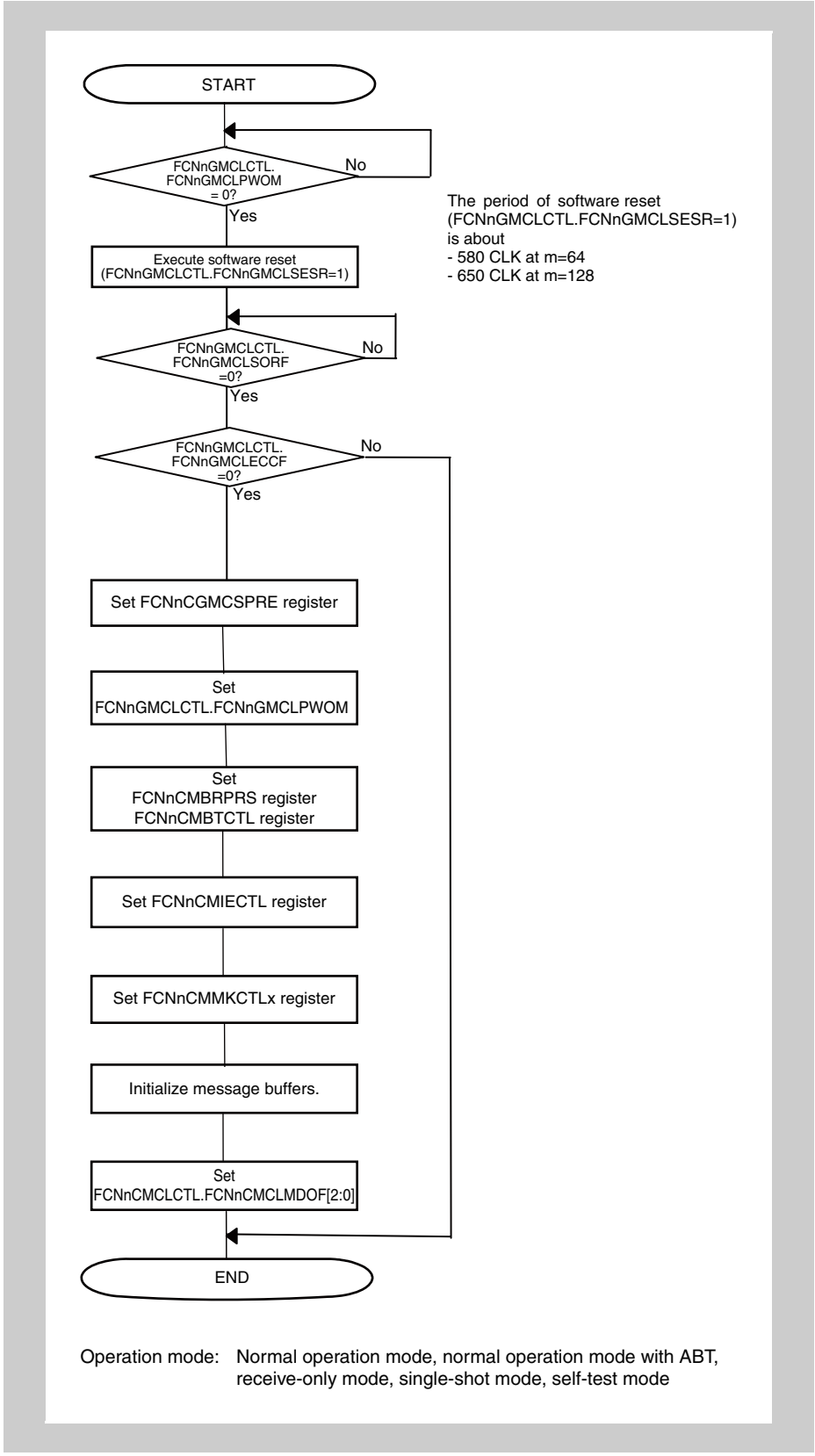


Figure 22-16 Re-initialization with Software Reset function

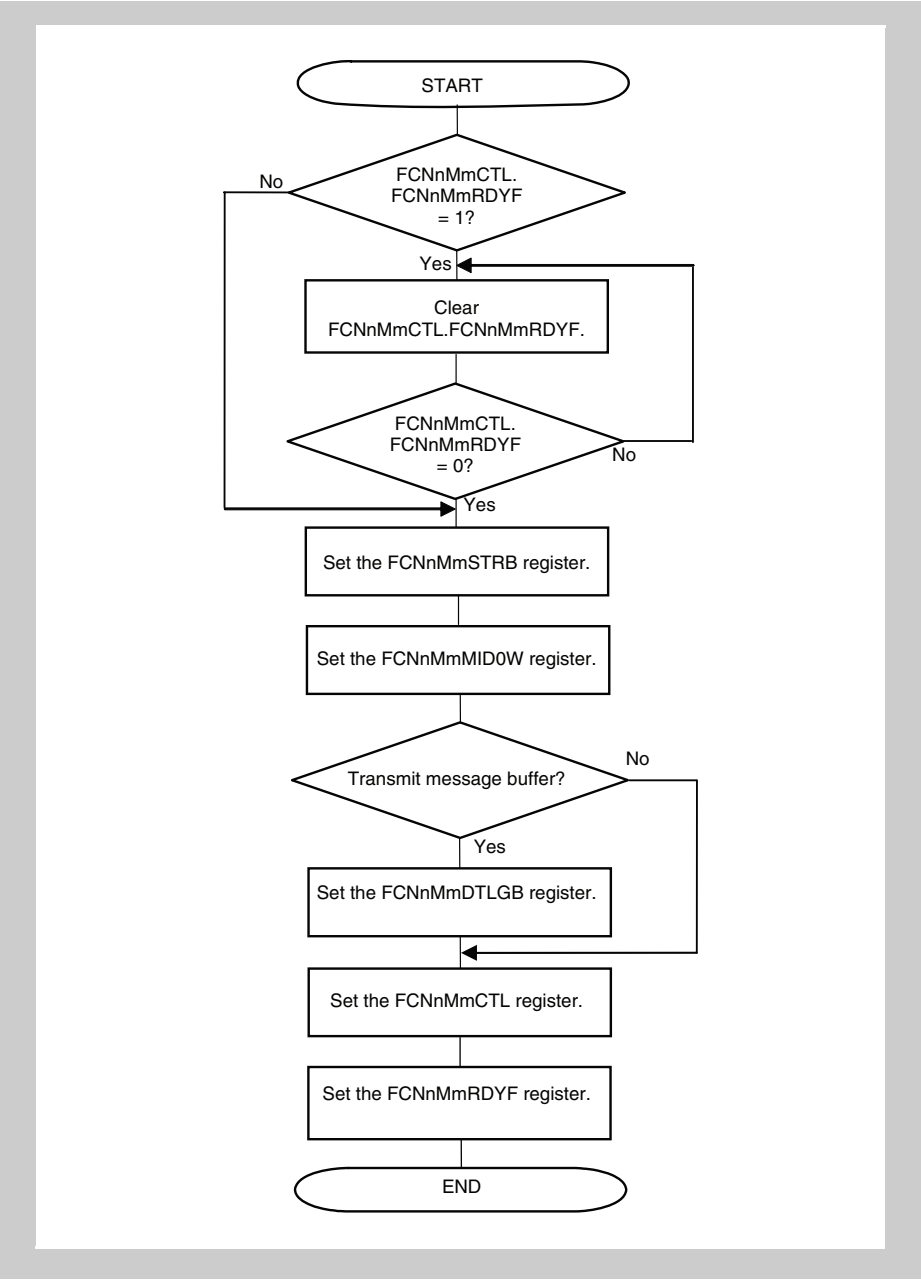
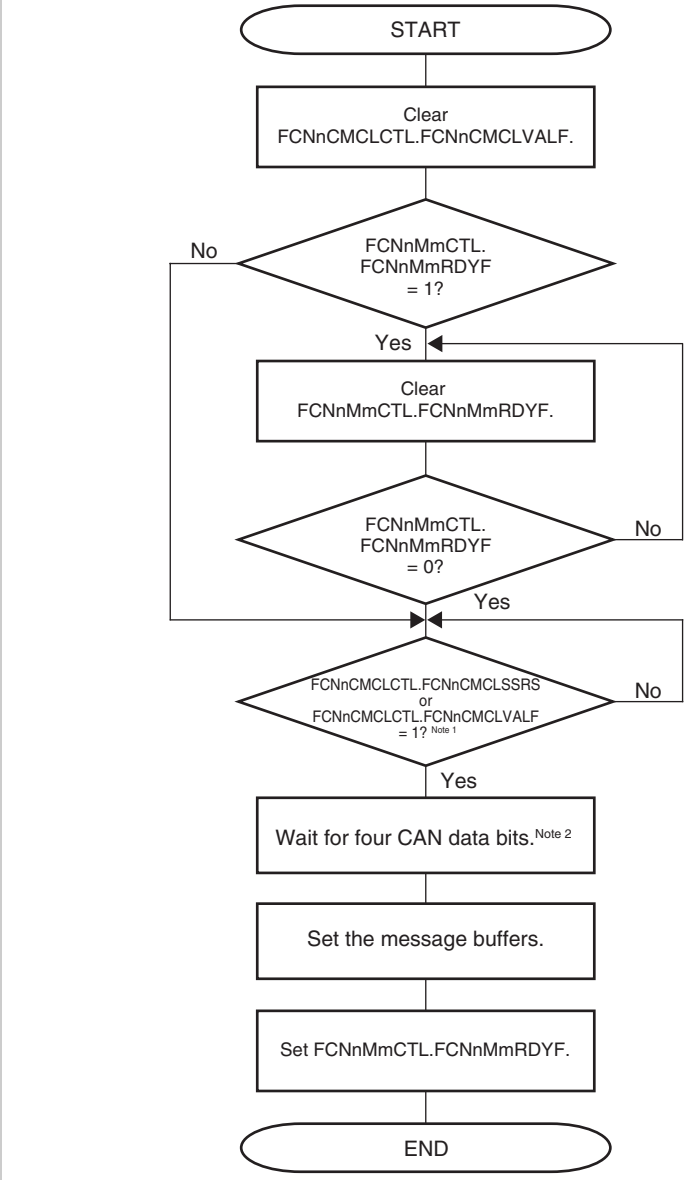


Figure 22-17 Message buffer initialization

Cautions

1. Before a message buffer is initialized, FCNnMmCTL.FCNnMmRDYF must be cleared.
2. Make the following settings for message buffers not used by the application.
   - Clear FCNnMmRDYF, FCNnMmTRQF, and FCNnMmDTNF bits of the FCNnMmCTL register to 0.
   - Clear FCNnMmSTRB.FCNnMmSSAM to 0.

Figure 22-18 "Message buffer redefinition during reception" shows the processing for a receive message buffer (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0001_B to 1000_B).



Notes 1. Confirm that a message is being received. It is necessary to set FCNmMmCTL.FCNnMmRDFY after a message is completely received.

2. Wait for four CAN data bits so that message buffers are not redefined while saving the received messages.

Figure 22-18 Message buffer redefinition during reception

Figure 22-19 "Message buffer redefinition during transmission" shows the processing for a transmit message buffer during transmission (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B).

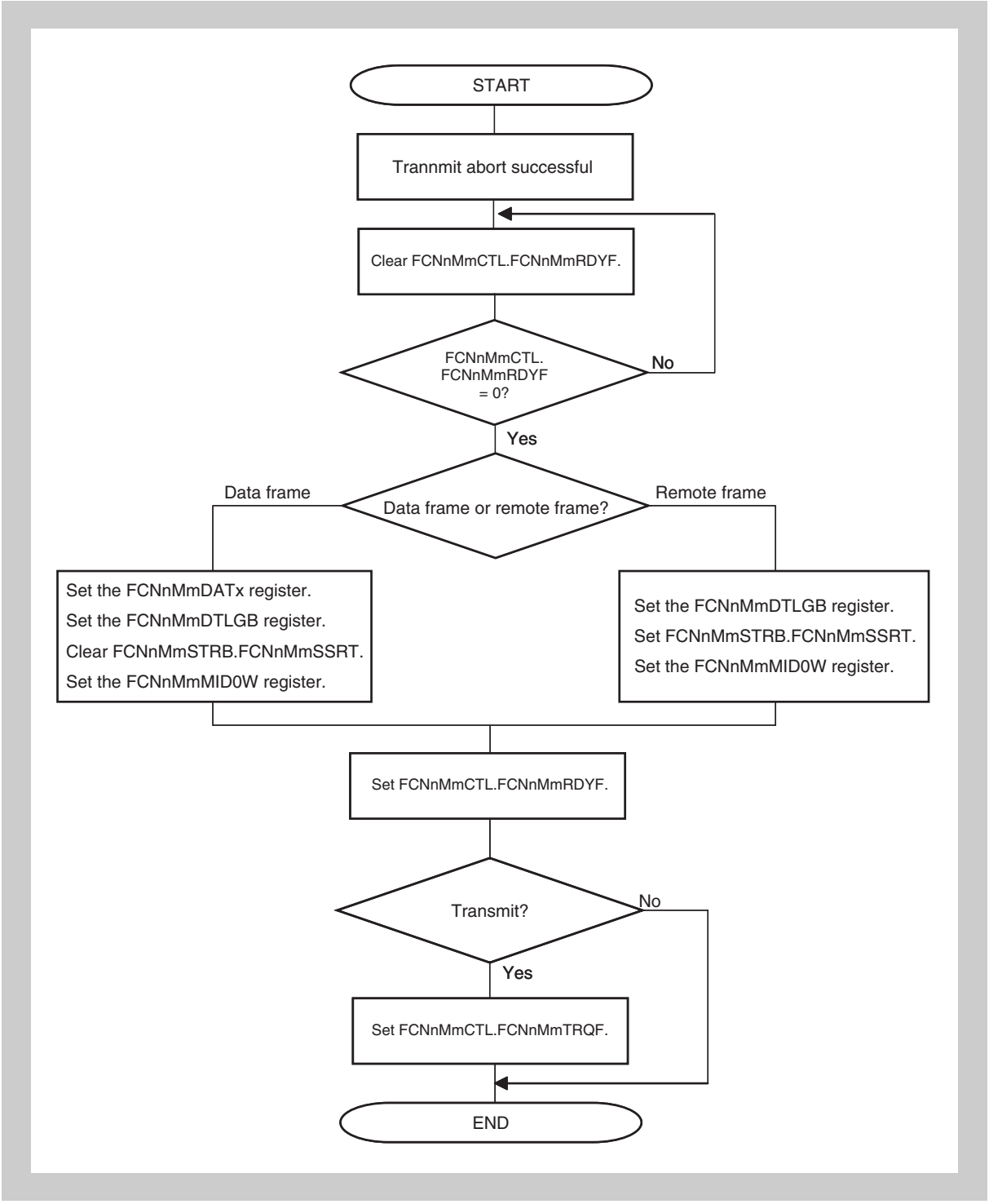
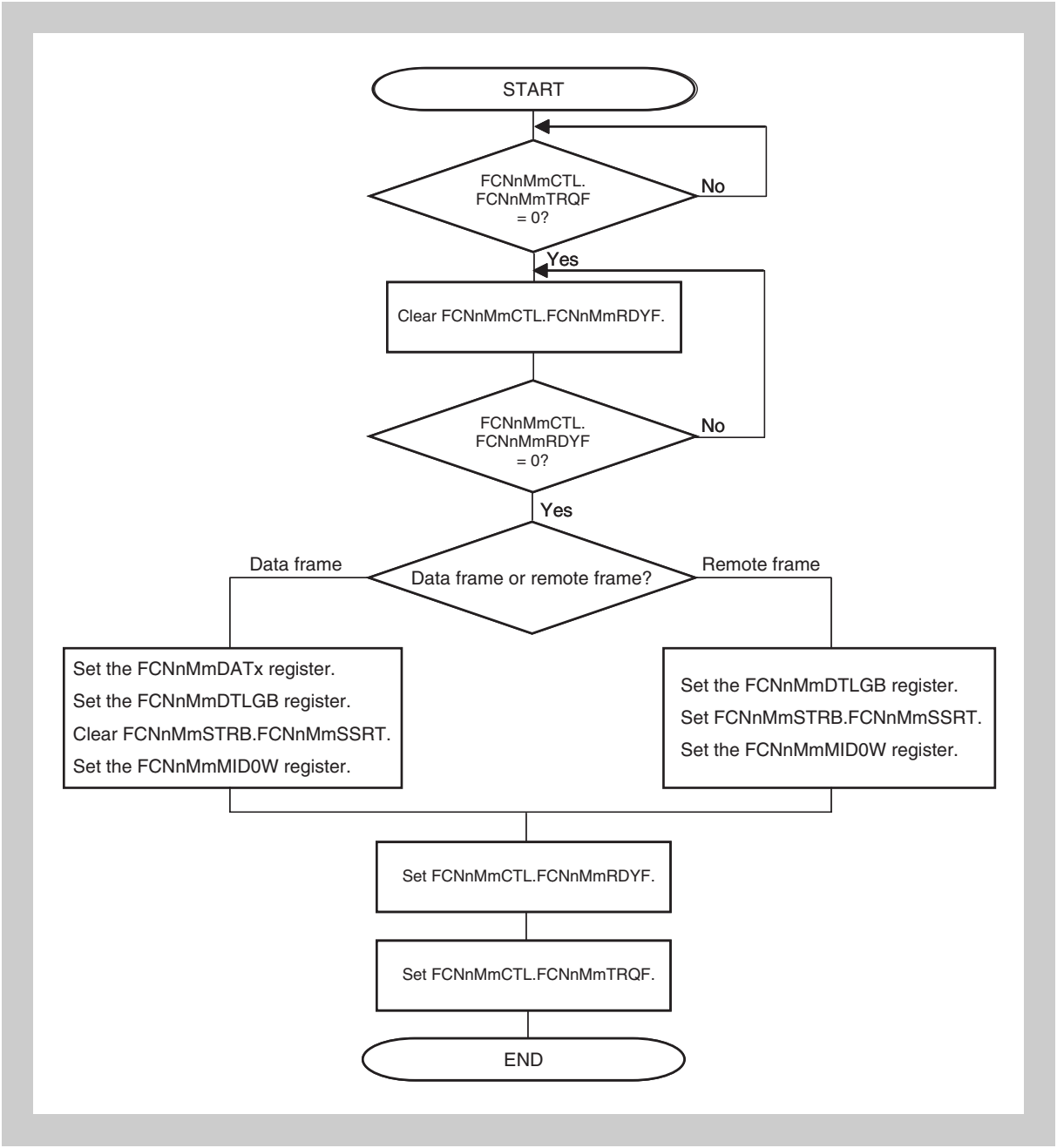


Figure 22-19 Message buffer redefinition during transmission

22.15.2 Message transmission

Figure 22-20 "Message transmit processing" shows the processing for a transmit message buffer (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B).



<R>

Figure 22-20 Message transmit processing

Cautions

1. FCNnMmCTL.FCNnMmTRQF should be set after FCNnMmCTL.FCNnMmRDYF is set.
2. FCNnMmCTL.FCNnMmRDYF and FCNnMmCTL.FCNnMmTRQF should not be set at the same time.

Figure 22-21 "ABT message transmit processing" shows the processing for a transmit message buffer (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B)

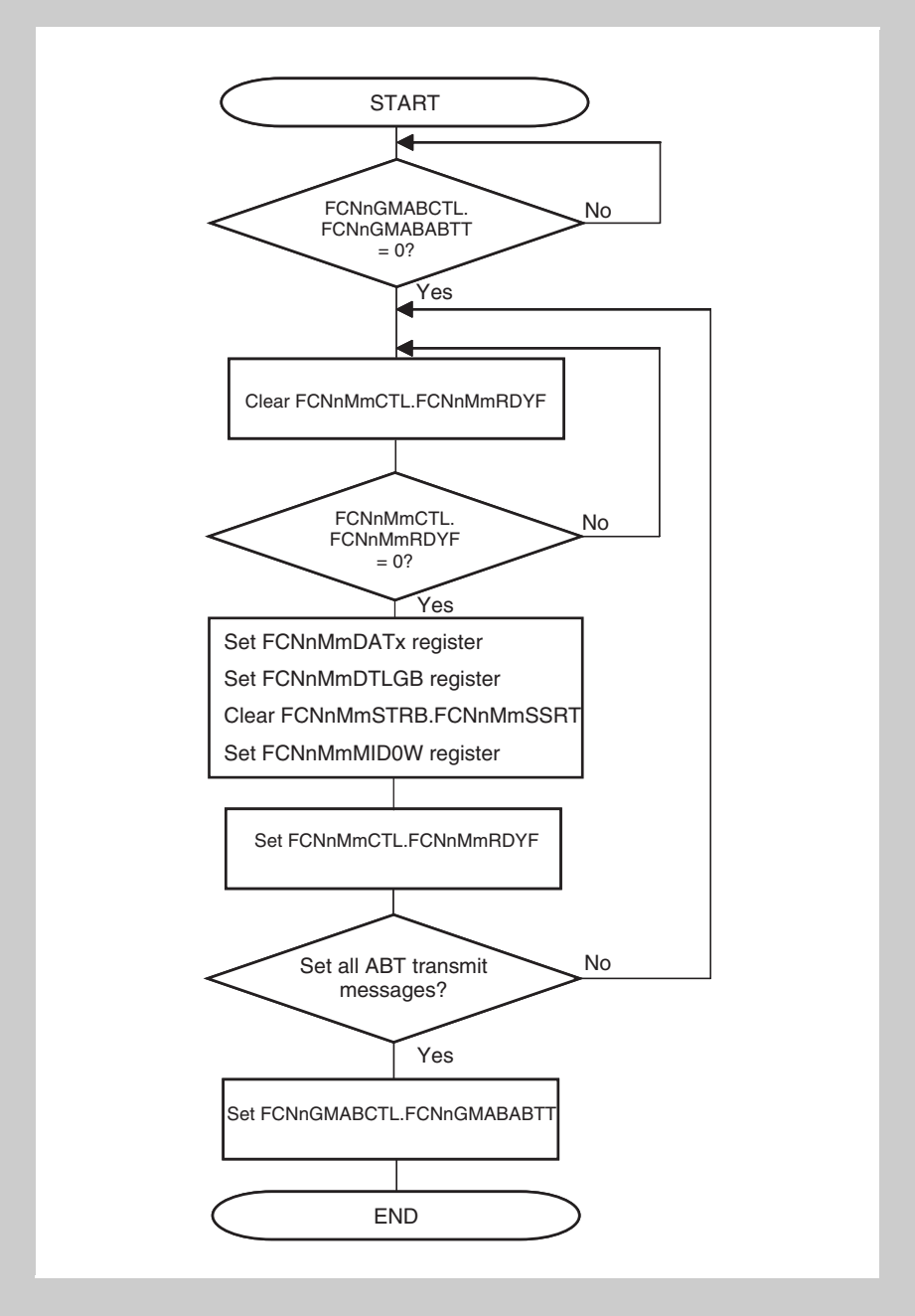


Figure 22-21 ABT message transmit processing

Note This processing (normal operation mode with ABT) can only be applied to message buffers usable with ABT mode. For message buffers other than the ABT message buffers, see *Figure 22-20 "Message transmit processing" on page 1576*.

<R>

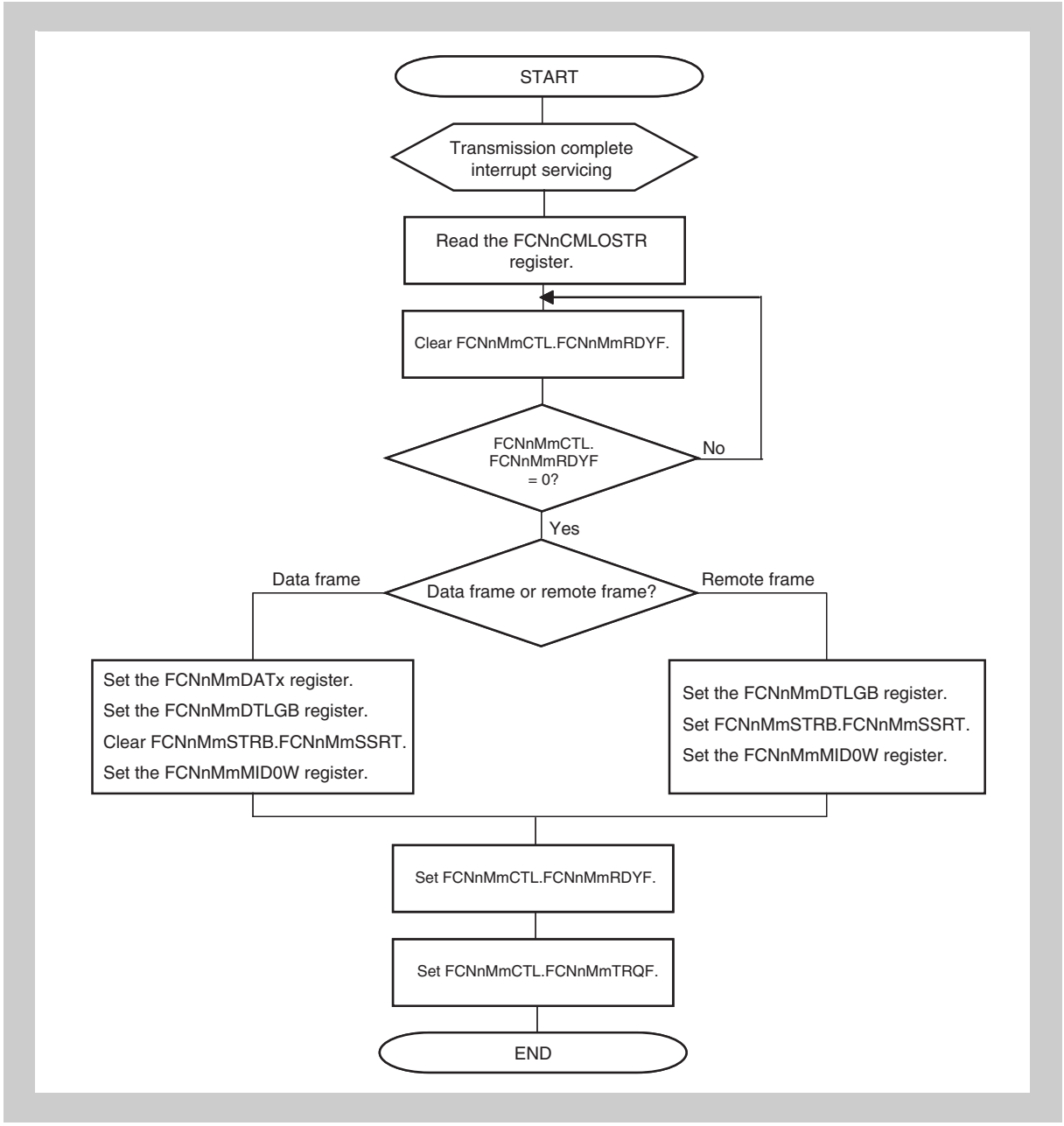


Figure 22-22 Transmission via interrupt (using FCNnCMLOSTR register)

Cautions

1. FCNnMmCTL.FCNnMmTRQF should be set after FCNnMmCTL.FCNnMmRDYF is set.
2. FCNnMmCTL.FCNnMmRDYF and FCNnMmCTL.FCNnMmTRQF should not be set at the same time.

Note Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as TX history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and

results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
It is recommended to cancel any sleep mode requests, before processing TX interrupts.

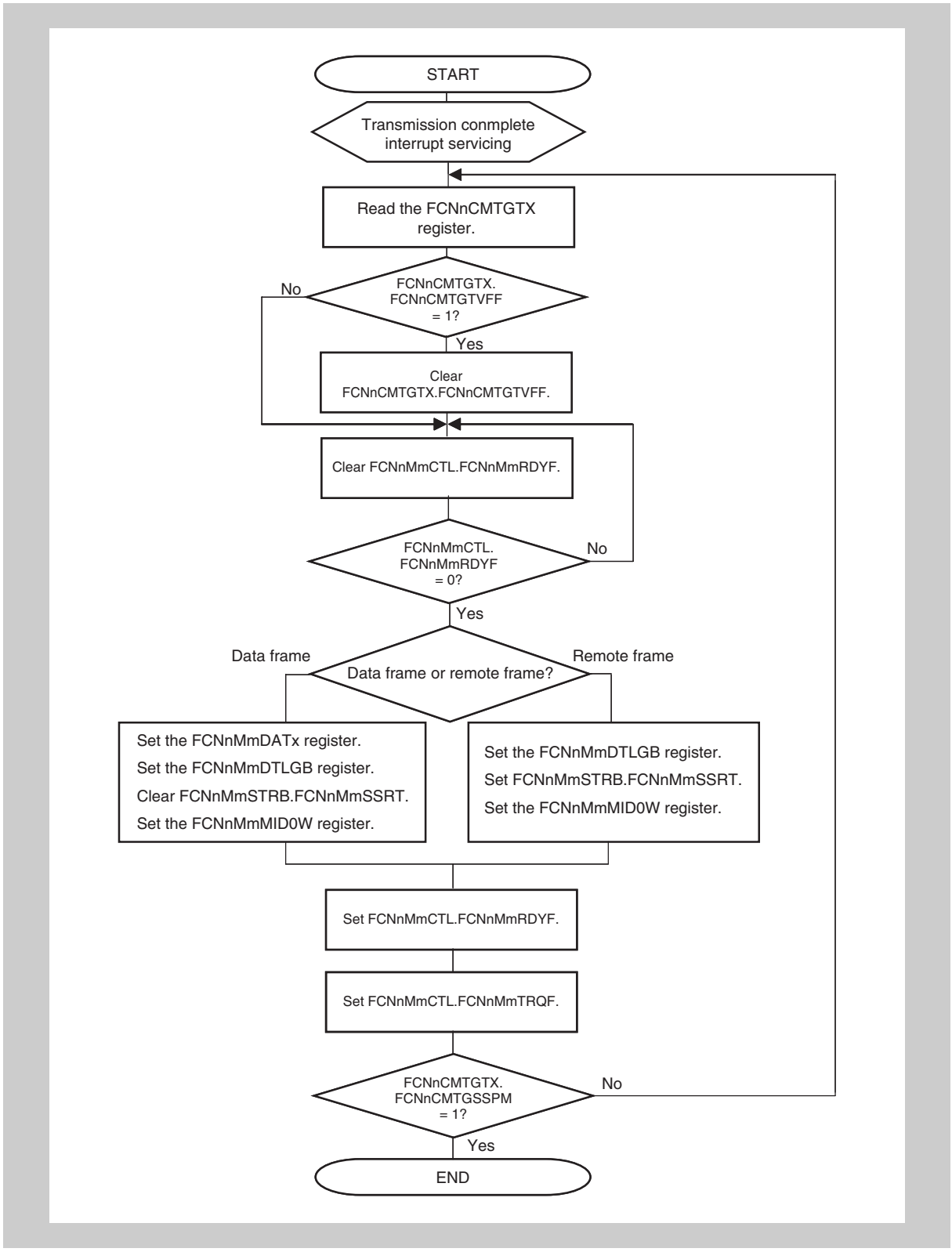


Figure 22-23 Transmission via interrupt (using FCNnCMTGTX register)

Cautions

1. FCNnMmCTL.FCNnMmTRQF should be set after FCNnMmCTL.FCNnMmRDYF is set.
2. FCNnMmCTL.FCNnMmRDYF and FCNnMmCTL.FCNnMmTRQF should not be set at the same time.

Notes

1. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as TX history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
   It is recommended to cancel any sleep mode requests, before processing TX interrupts.
2. If FCNnCMTGTX.FCNnCMTGTVFF was set once, the transmit history list is inconsistent. Consider to scan all configured transmit buffers for completed transmissions.

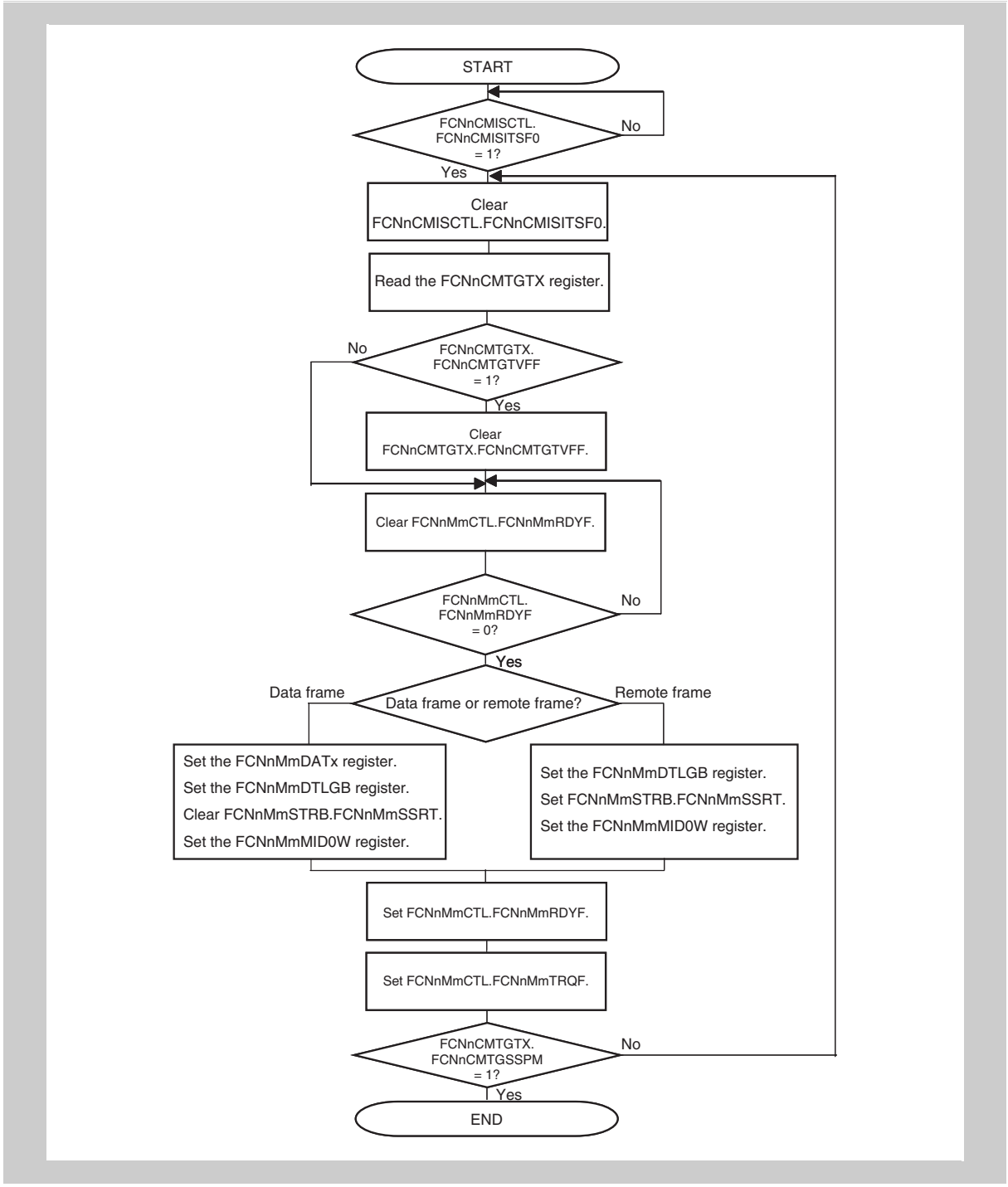


Figure 22-24 Transmission via software polling

Cautions

1. FCNnMmCTL.FCNnMmTRQF should be set after FCNnMmCTL.FCNnMmRDYF is set.
2. FCNnMmCTL.FCNnMmRDYF and FCNnMmCTL.FCNnMmTRQF should not be set at the same time.

Notes

1. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the polling routine, in order to check the access to the message buffers as well as TX history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
2. If FCNnCMTGTX.FCNnCMTGTVFF was set once, the transmit history list is inconsistent. Consider to scan all configured transmit buffers for completed transmissions.

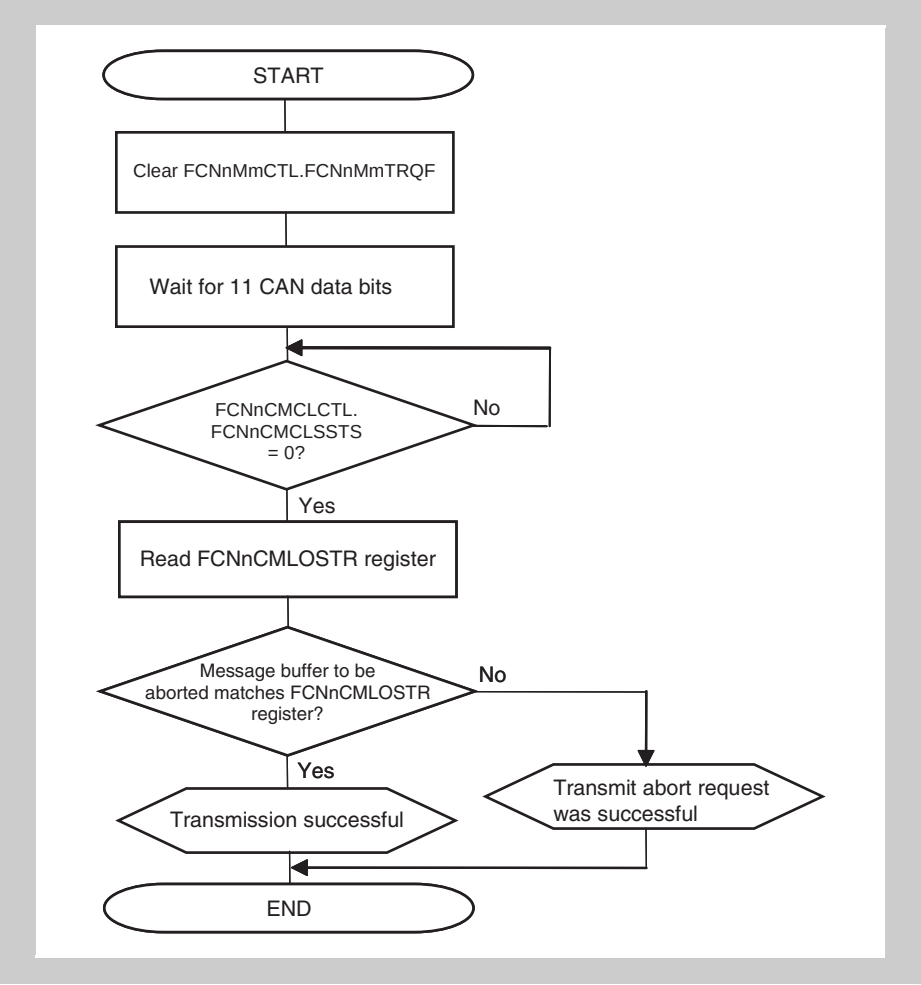
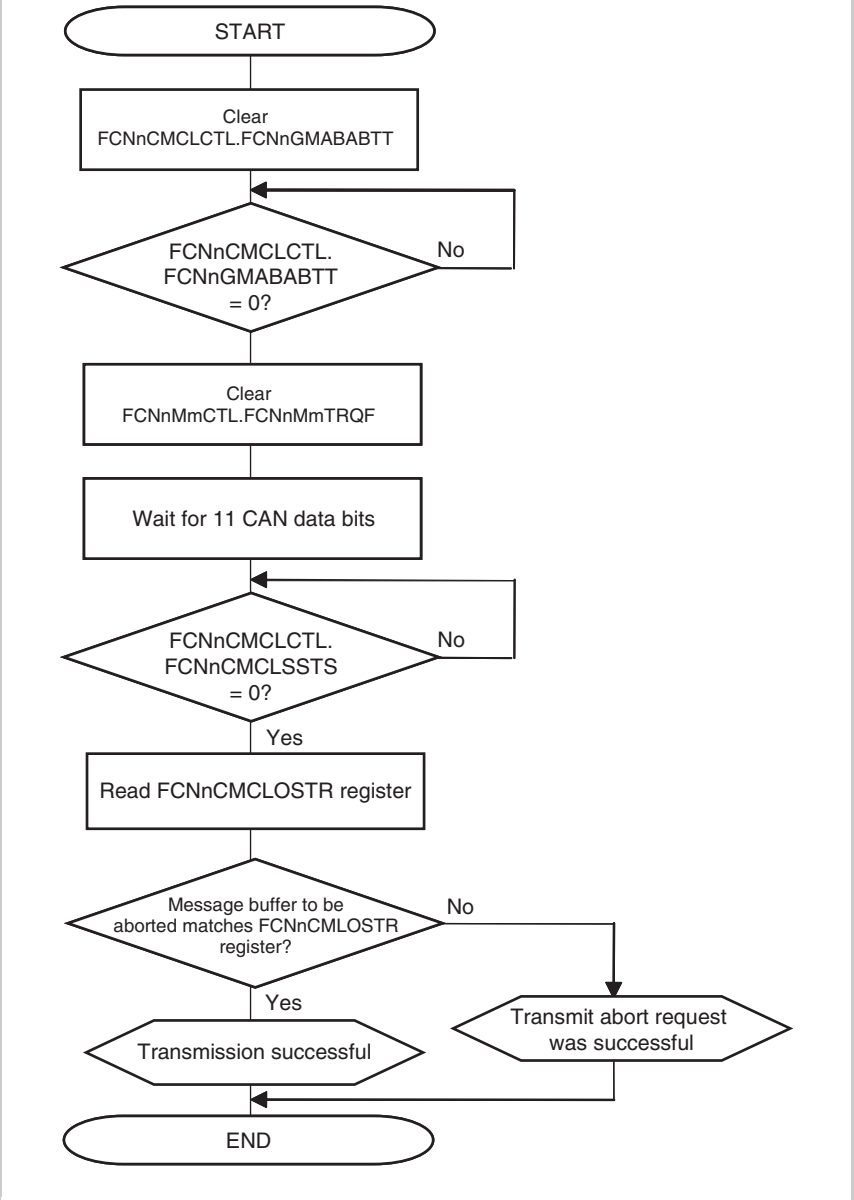


Figure 22-25 Transmission abort processing (except normal operation mode with ABT)

Cautions

1. Clear FCNnMmCTL.FCNnMmTRQF for aborting transmission request, not FCNnMmCTL.FCNnMmRDYF.
2. Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.
3. FCNnCMCLCTL.FCNnCMCLSSTS can be periodically checked by a user application or can be checked after the transmit completion interrupt.
4. Do not execute any new transmission request including in the other message buffers while transmission abort processing is in progress.

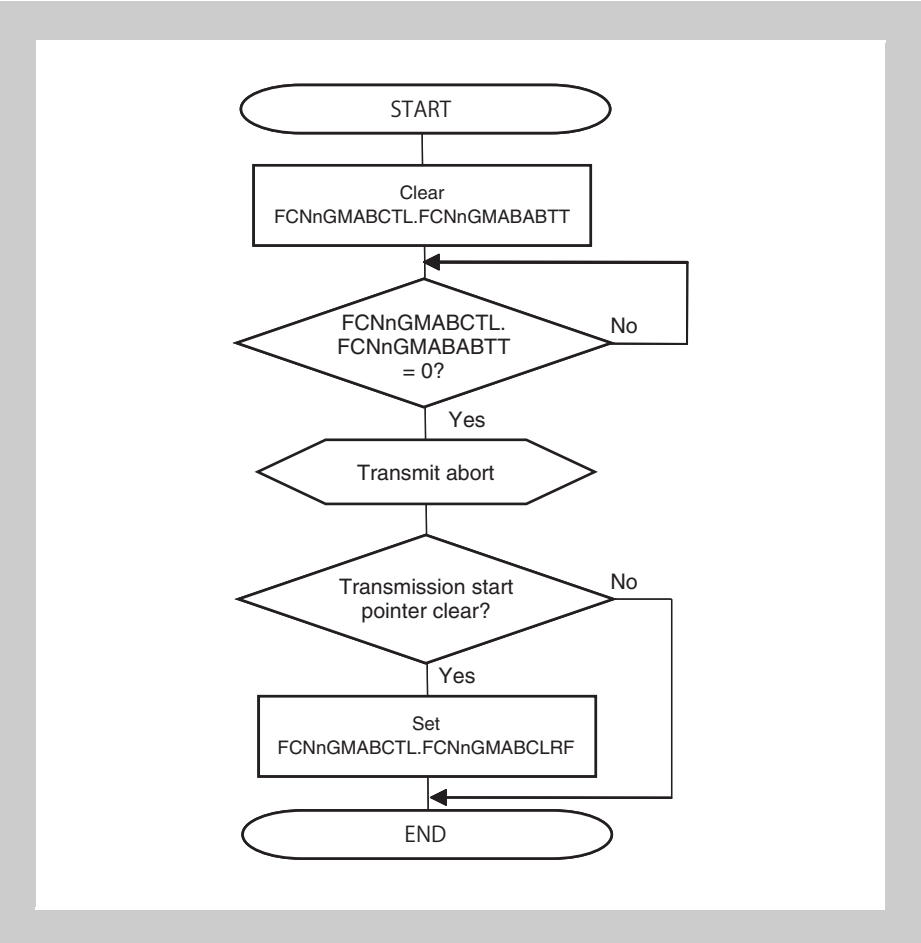


<R> **Figure 22-26 Processing to abort transmission other than ABT transmission (in normal operation mode with ABT)**

Cautions

1. Clear FCNnMmCTL.FCNnMmTRQF for aborting transmission request, not FCNnMmCTL.FCNnMmRDYF.
2. Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.
3. FCNnCMCLCTL.FCNnCMCLSSTS can be periodically checked by a user application or can be checked after the transmit completion interrupt.
4. Do not execute any new transmission request including in the other message buffers while transmission abort processing is in progress.

Figure 22-27 "ABT transmission request abort processing (in normal operation mode with ABT) (1)" shows the processing to not skip resumption of transmitting a message that was stopped when transmission of an ABT message buffer was aborted.



<R> **Figure 22-27 ABT transmission request abort processing (in normal operation mode with ABT) (1)**

<R> **Cautions**

1. Do not set any transmission requests while ABT transmission abort processing is in progress.

<R> 2. Make a FCN sleep mode/FCN stop mode request after FCNnGMABCTL.FCNnGMABABTT is cleared (after ABT mode is stopped) following the procedure shown in *Figure 22-27 "ABT transmission request abort processing (in normal operation mode with ABT) (1)"*. When clearing a transmission request in an area other than the ABT area, follow the procedure shown in *Figure 22-25 "Transmission abort processing (except normal operation mode with ABT)" on page 1582*.

Figure 22-28 "ABT transmission request abort processing (in normal operation mode with ABT) (2)" shows the processing to not skip resumption of transmitting a message that was stopped when transmission of an ABT message buffer was aborted.

<R>

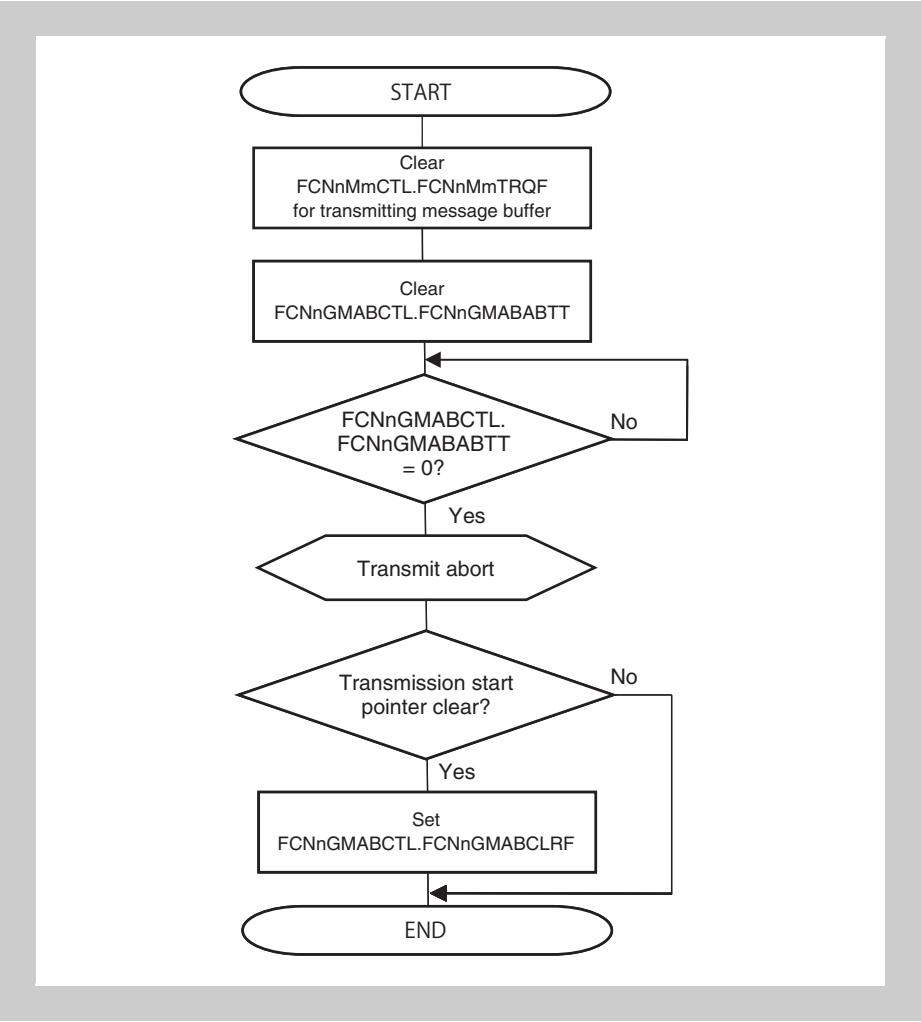


Figure 22-28 ABT transmission request abort processing (in normal operation mode with ABT) (2)

Cautions

1. Do not set any transmission requests while ABT transmission abort processing is in progress.

<R>

2. Make a FCN sleep mode/FCN stop mode request after FCNnGMABCTL.FCNnGMABABTT is cleared (after ABT mode is stopped) following the procedure shown in *Figure 22-27 "ABT transmission request abort processing (in normal operation mode with ABT) (1)"*. When clearing a transmission request in an area other than the ABT area, follow the procedure shown in *Figure 22-25 "Transmission abort processing (except normal operation mode with ABT)" on page 1582*.

Figure 22-29 "ABT transmission request abort processing (normal operation mode with ABT) with transmission completely finished flag" shows the processing on ABT mode, when using the Transmit Abort functionality (transmit flag). The box "Transmission abort success" represents the checking of the transmission abort success by checking the FCNnMmTCPF flag within the ABT message buffers.

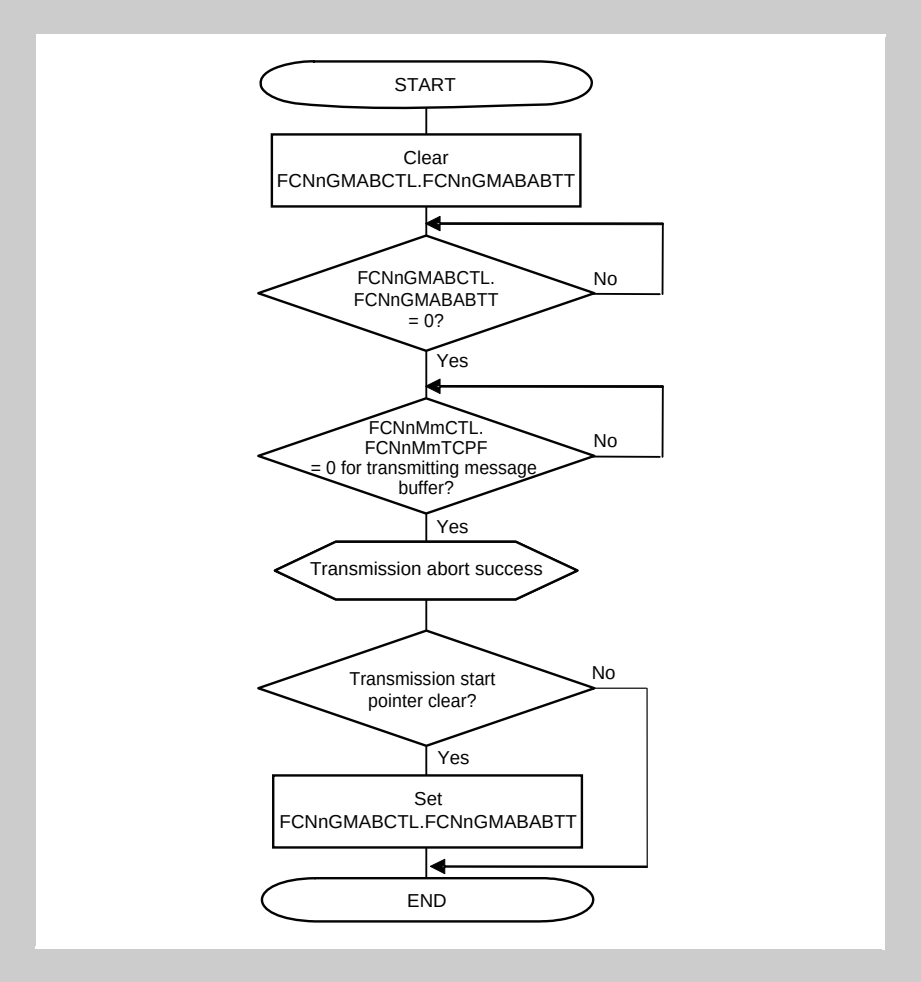


Figure 22-29 ABT transmission request abort processing (normal operation mode with ABT) with transmission completely finished flag

Cautions

1. Do not set any transmission requests while ABT transmission abort processing is in progress.
2. Make a FCN sleep mode/FCN stop mode request after FCNnGMABCTL.FCNnGMABABTT is cleared (after ABT mode is stopped) following the procedure shown in *Figure 22-27 "ABT transmission request abort processing (in normal operation mode with ABT) (1)"*. When clearing a transmission request in an area other than the ABT area, follow the procedure shown in *Figure 22-25 "Transmission abort processing (except normal operation mode with ABT)" on page 1582*.

Note There is the case that all ABT is transmitted completely even if ABT transmission abort processing is performed successfully. Then it is possible to know which message is finished transmission.

Figure 22-30 "Transmission request abort processing with transmit abort interrupt and transmission completely finished flag" shows the processing when using the Transmit Abort functionality (Transmit Abort Interrupt).

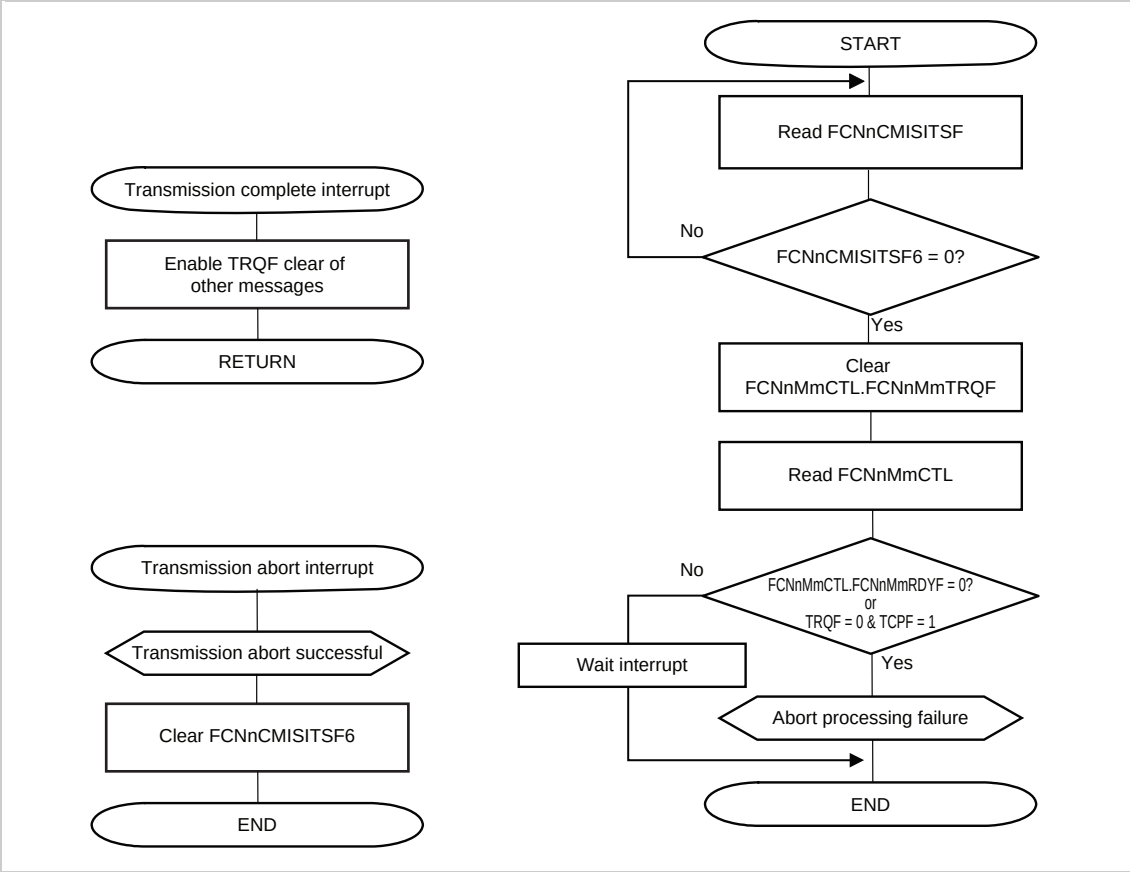


Figure 22-30 Transmission request abort processing with transmit abort interrupt and transmission completely finished flag

Note The determination that FCNnMmRDYF=0 is made with consideration for the case when FCNnMmRDYF is cleared during transmission completion processing due to an interrupt.

Cautions

1. Abort transmissions by clearing FCNnMmTRQF rather than by clearing FCNnMmRDYF.
2. Make sure that the transmission request made via this flow sequence has completely ended before sending a sleep request.
3. Do not update messages subject to transmission abort processing (FCNnMmRDYF or FCNnMmTRQF is set), for example via transmission complete interrupt processing.
4. Do not clear FCNnMmTRQF in other message buffers while performing transmission abort processing.
5. If you change the ID to one with lower priority during transmission abort processing, then wait for at least one frame after clearing FCNnMmTRQF before sending a transmission request.

<R>

6. Always read FCNnMmTRQF and FCNnMmTCPF at least once.

Figure 22-31 "Transmission request abort processing with transmission completely finished flag" shows the processing when using the Transmit Abort functionality (Transmission Completely Finished Flag FCNnMmTCPF).

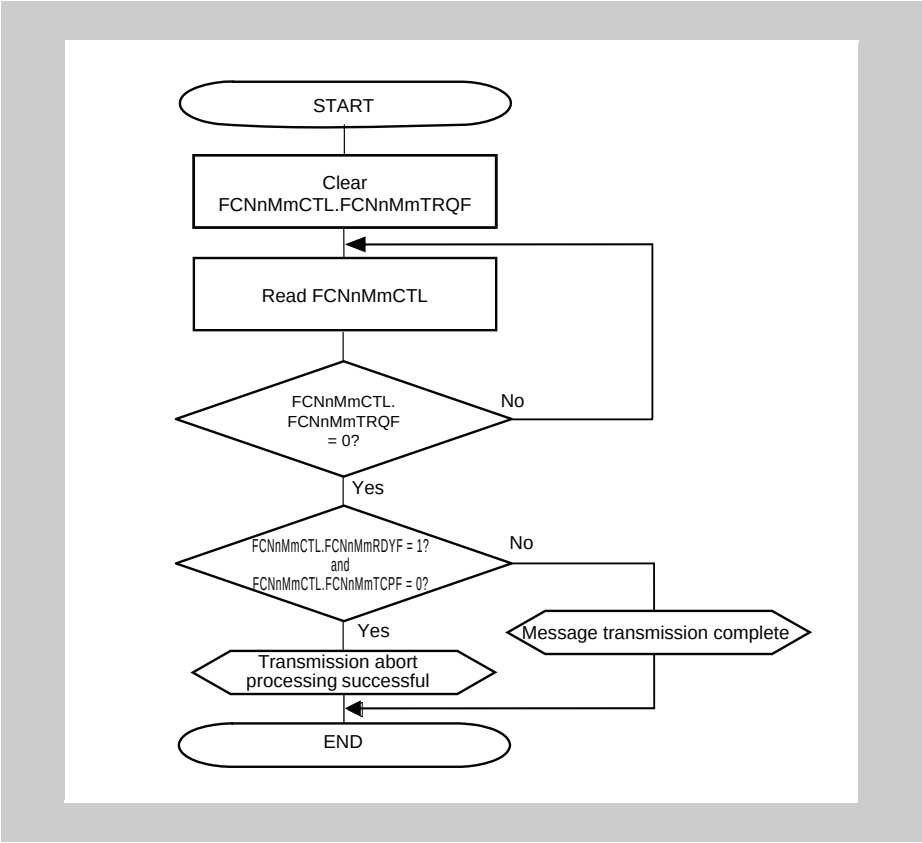


Figure 22-31 Transmission request abort processing with transmission completely finished flag

Note The determination that FCNnMmRDYF=0 is made with consideration for the case when FCNnMmRDYF is cleared during transmission completion processing due to an interrupt.

Cautions

1. Abort transmissions by clearing FCNnMmTRQF rather than by clearing FCNnMmRDYF.
2. Make sure that the transmission request made via this flow sequence has completely ended before sending a sleep request.
3. Do not update messages subject to transmission abort processing (FCNnMmRDYF or FCNnMmTRQF is set), for example via transmission complete interrupt processing.
4. If you change the ID to one with lower priority during transmission abort processing, then wait for at least one frame after clearing FCNnMmTRQF before sending a transmission request.
5. Always read FCNnMmTRQF and FCNnMmTCPF at least once.

22.15.3 Message reception

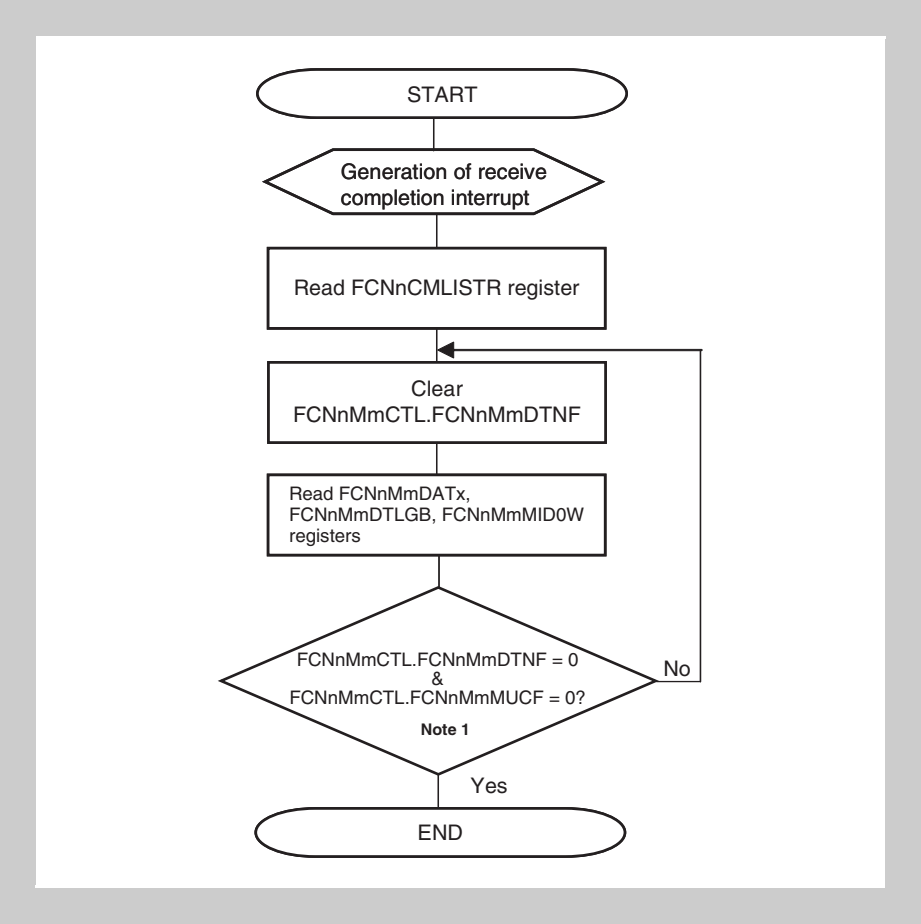


Figure 22-32 Reception via interrupt (using FCNnCMLISTR register)

Notes

1. Check FCNnMmCTL.FCNnMmMUCF and FCNnMmCTL.FCNnMmDTNF bits using one read access.
2. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as reception history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
   It is recommended to cancel any sleep mode requests, before processing RX interrupts.

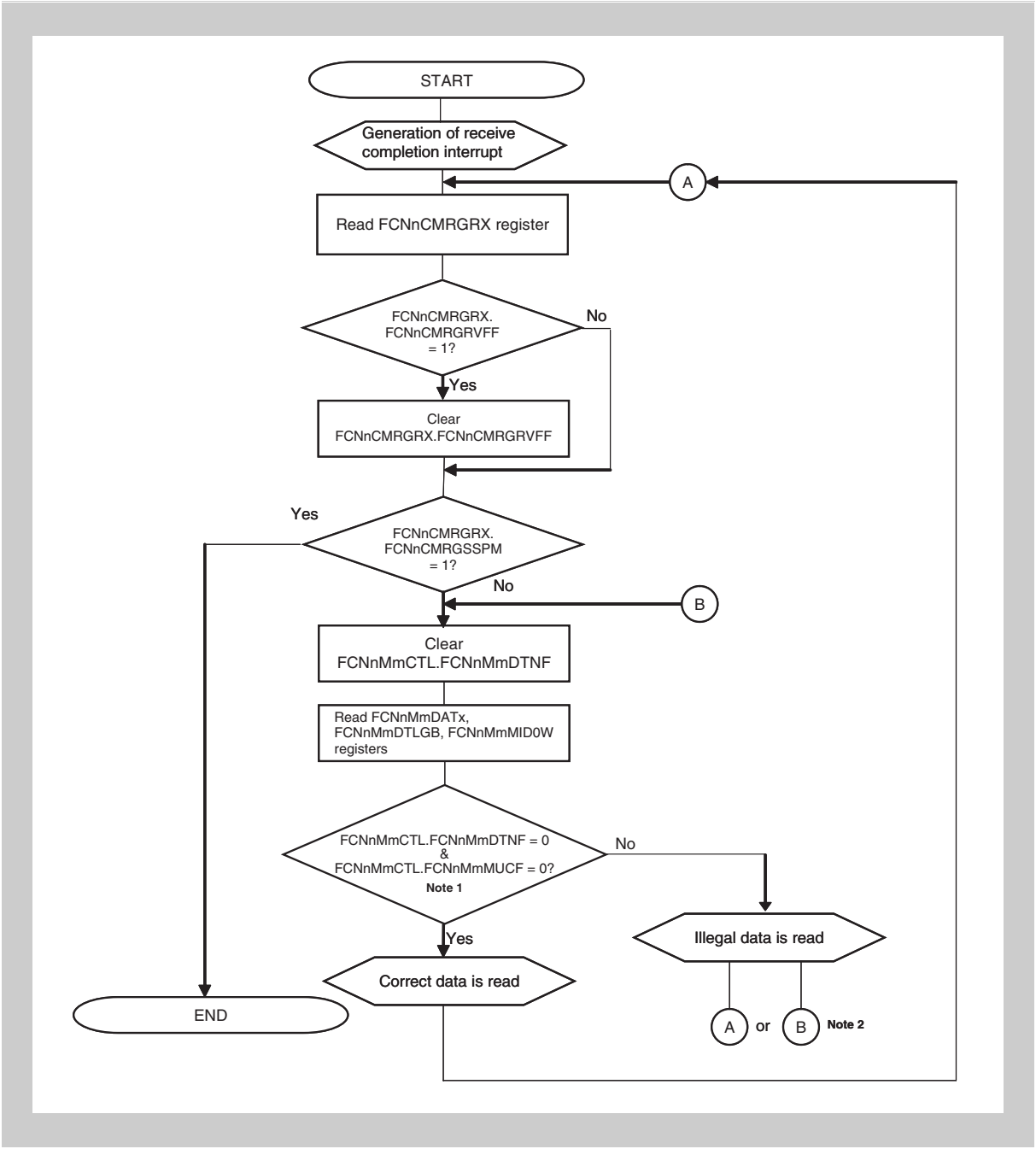


Figure 22-33 Reception via interrupt (using FCNnCMRGRX register)

Notes

1. Check FCNnMmCTL.FCNnMmMUCF and FCNnMmCTL.FCNnMmDTNF bits using one read access.
2. Depending of the processing target of the application, two ways are possible:
   –Way A: The message is not processed within this pass, but with the next pass, depending on the timing this can happen latest with the next Receive Interrupt.
   Other messages will be processed earlier.
   – Way B: The message is processed within this pass, the loop waits on this message.
   Other messages will be processed later.
3. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as reception history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
   It is recommended to cancel any sleep mode requests, before processing RX interrupts.
4. If FCNnCMRGRX.FCNnCMRGRVFF was set once, the receive history list is inconsistent. Consider to scan all configured receive buffers for receptions.

<R>

5. For the processing shown in *Figure 22-33 “Reception via interrupt (using FCNnCMRGRX register)”*, the processing shown in *Figure 22-34 “Reception via interrupt (using FCNnCMRGRX register), alternative way”* can be used alternatively.

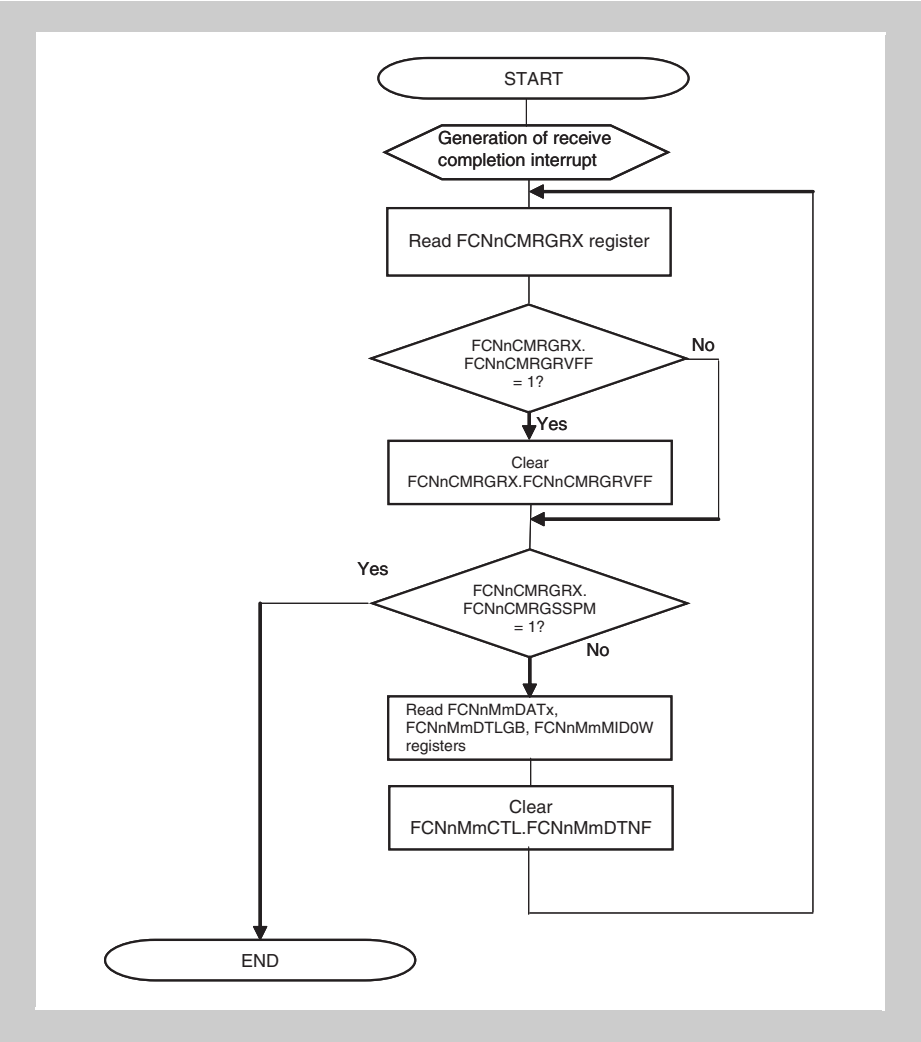


Figure 22-34 Reception via interrupt (using FCNnCMRGRX register), alternative way

Notes

1. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as reception history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
   It is recommended to cancel any sleep mode requests, before processing RX interrupts.
2. If FCNnCMRGRX.FCNnCMRGRVFF was set once, the receive history list is inconsistent. Consider to scan all configured receive buffers for receptions.
3. This flow will not provide most recently received data for the application. However, due to less effort on processing, it reduces interrupt load.
4. The overwrite function (FCNnMmSTRB.FCNnMmSSOW=1) must not be used with this flow - data inconsistency could occur.

<R>

<R>

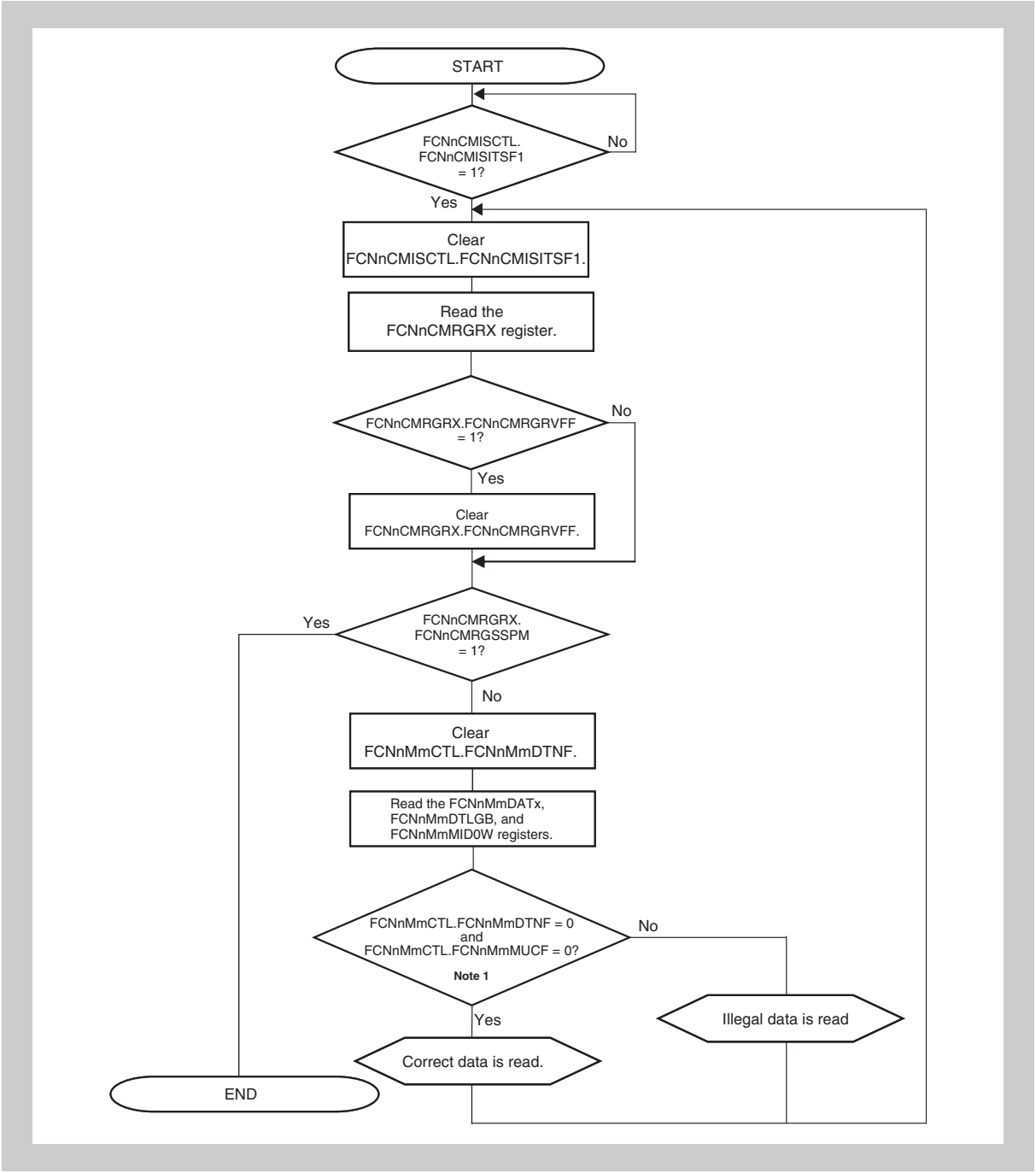


Figure 22-35 Reception via software polling

Notes

1. Check FCNnMmCTL.FCNnMmMUCF and FCNnMmCTL.FCNnMmDTNF bits using one read access.
2. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the polling routine, in order to check the access to the message buffers as well as reception history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
3. If FCNnCMRGRX.FCNnCMRGRVFF was set once, the receive history list is inconsistent. Consider to scan all configured receive buffers for receptions.

22.15.4 Power save modes

<R>

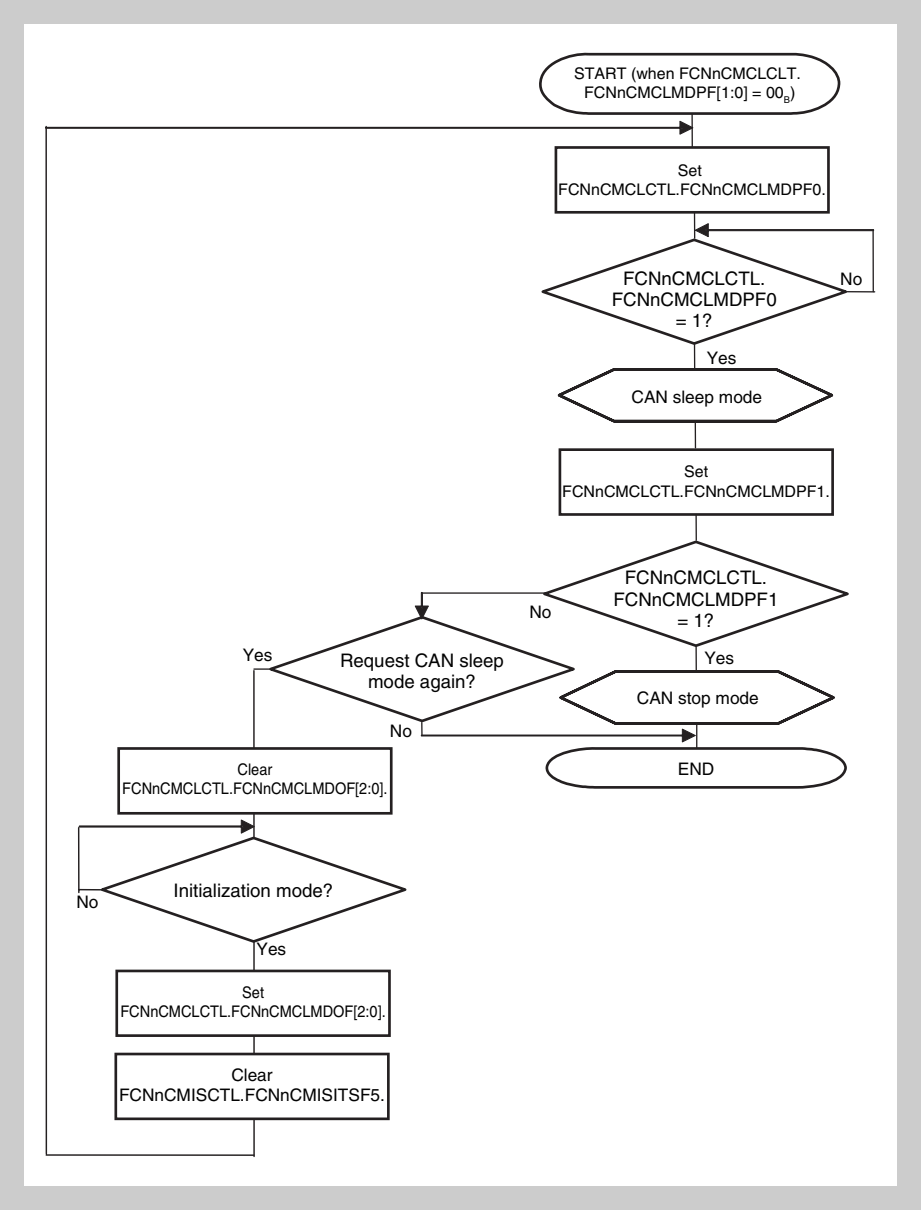


Figure 22-36 Setting FCN sleep mode/stop mode

Caution To abort transmission before making a request for the FCN sleep mode, perform processing according to previously given flowcharts.

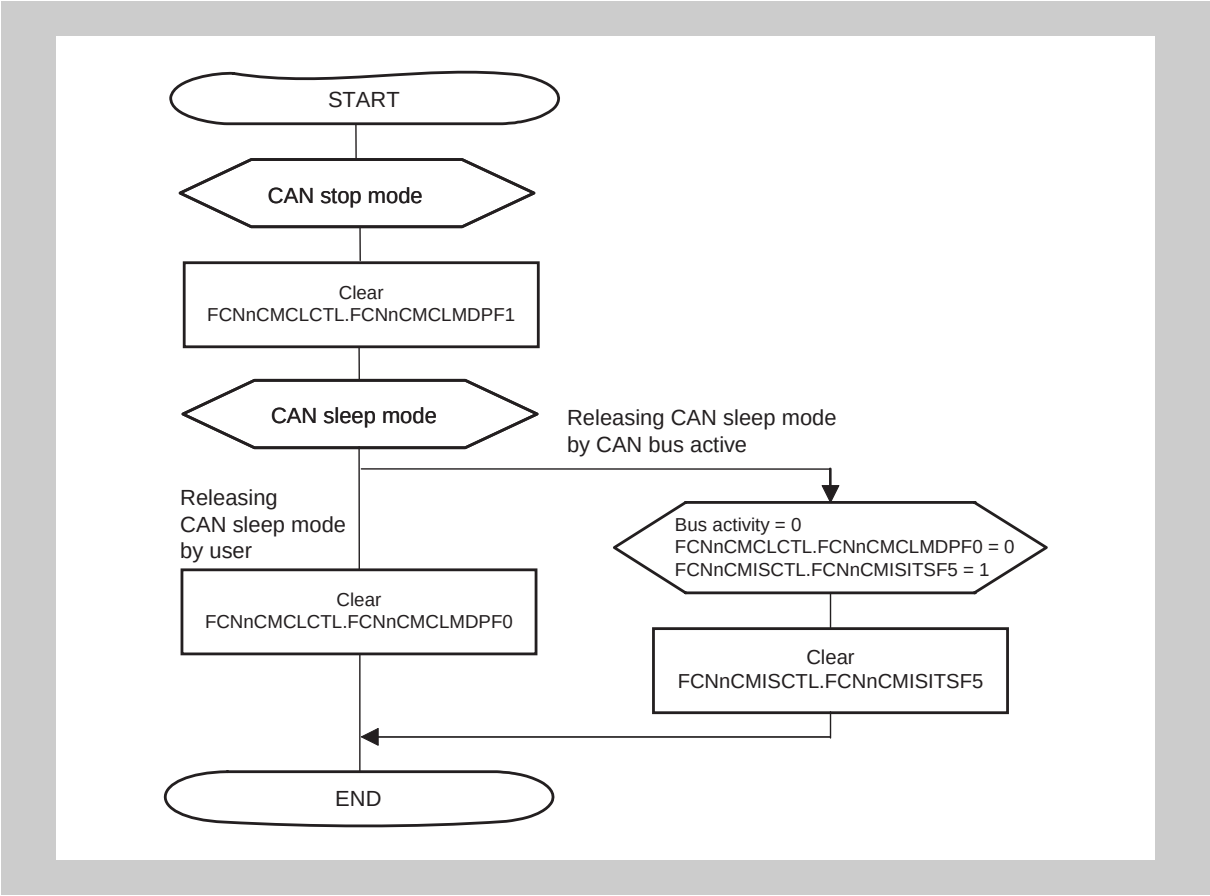
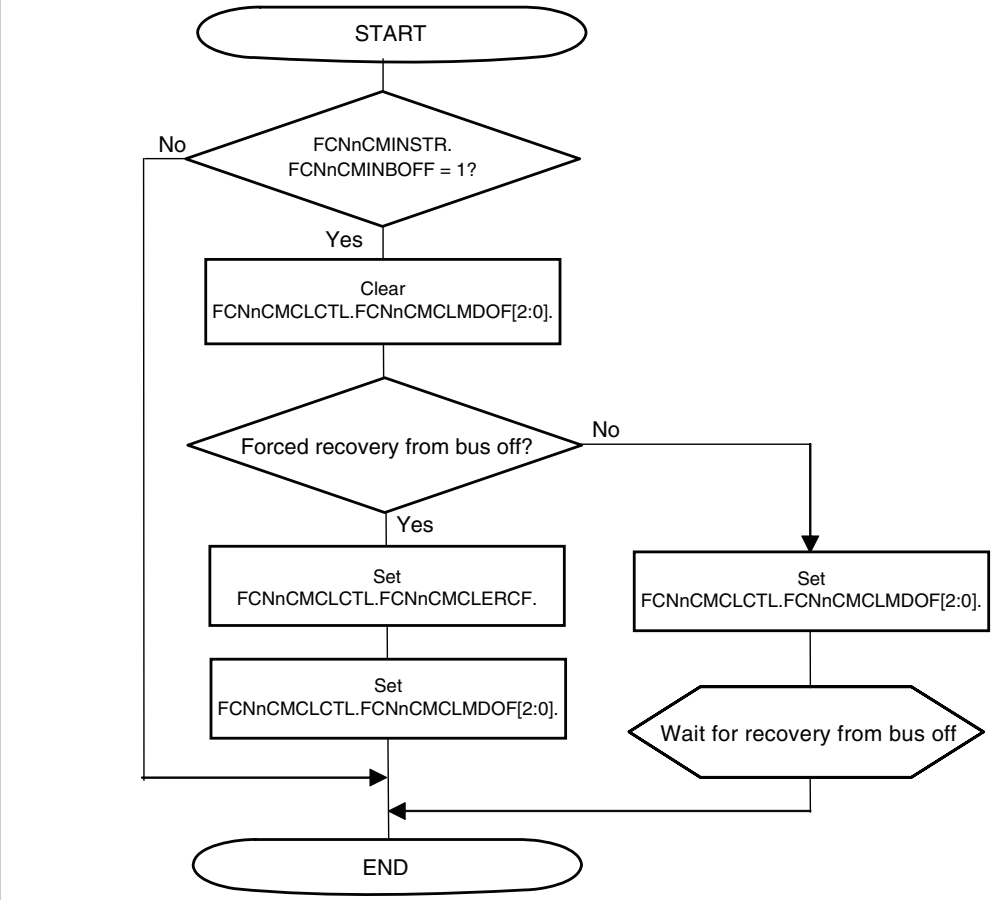


Figure 22-37 Clear FCN sleep/stop mode

<R>



<R> **Figure 22-38 Bus-off recovery**

Caution When the transmission from the initialization mode to any operation modes is requested to execute bus-off recovery sequence again in the bus-off recovery sequence, reception error counter is cleared.
Therefore it is necessary to detect 11 consecutive recessive-level bits 128 times on the bus again.

Note Operation mode: Normal operation mode, normal operation mode with ABT, receive-only mode, singleshot mode, self-test mode.

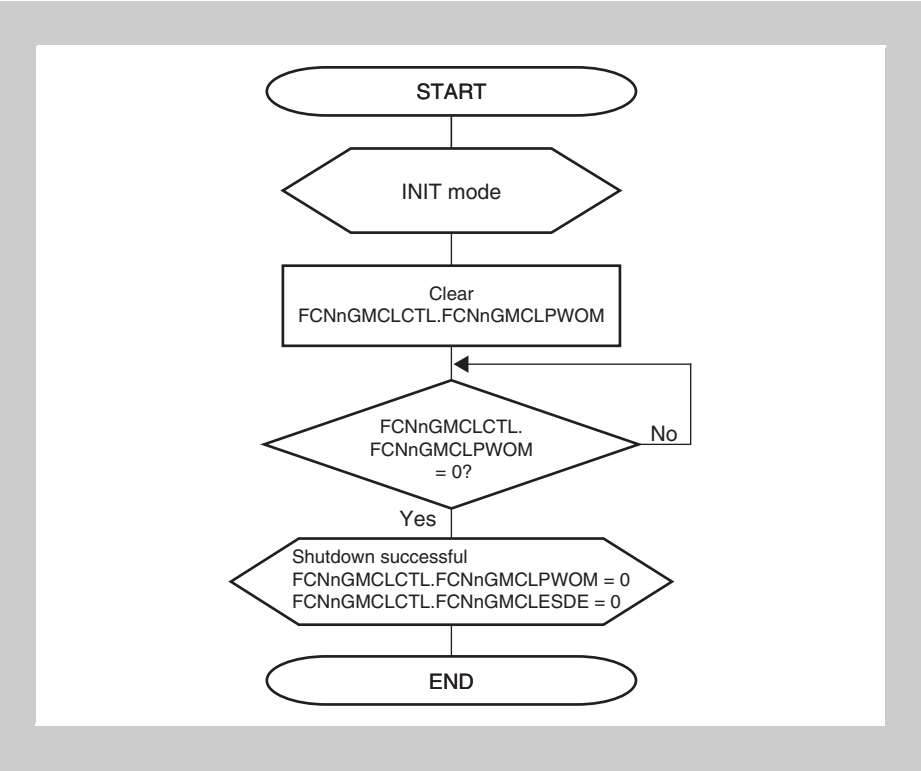


Figure 22-39 Normal shutdown process

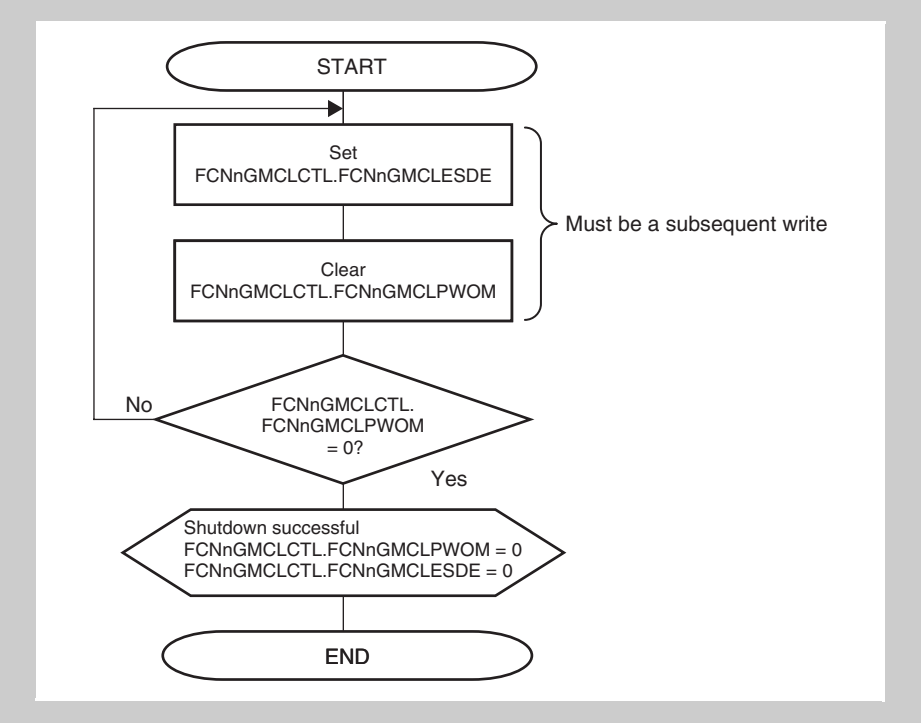


Figure 22-40 Forced shutdown process

Caution Do not read- or write-access any registers by software between setting the FCNnGMCLESDE bit and clearing the FCNnGMCLPWOM bit.

<R>

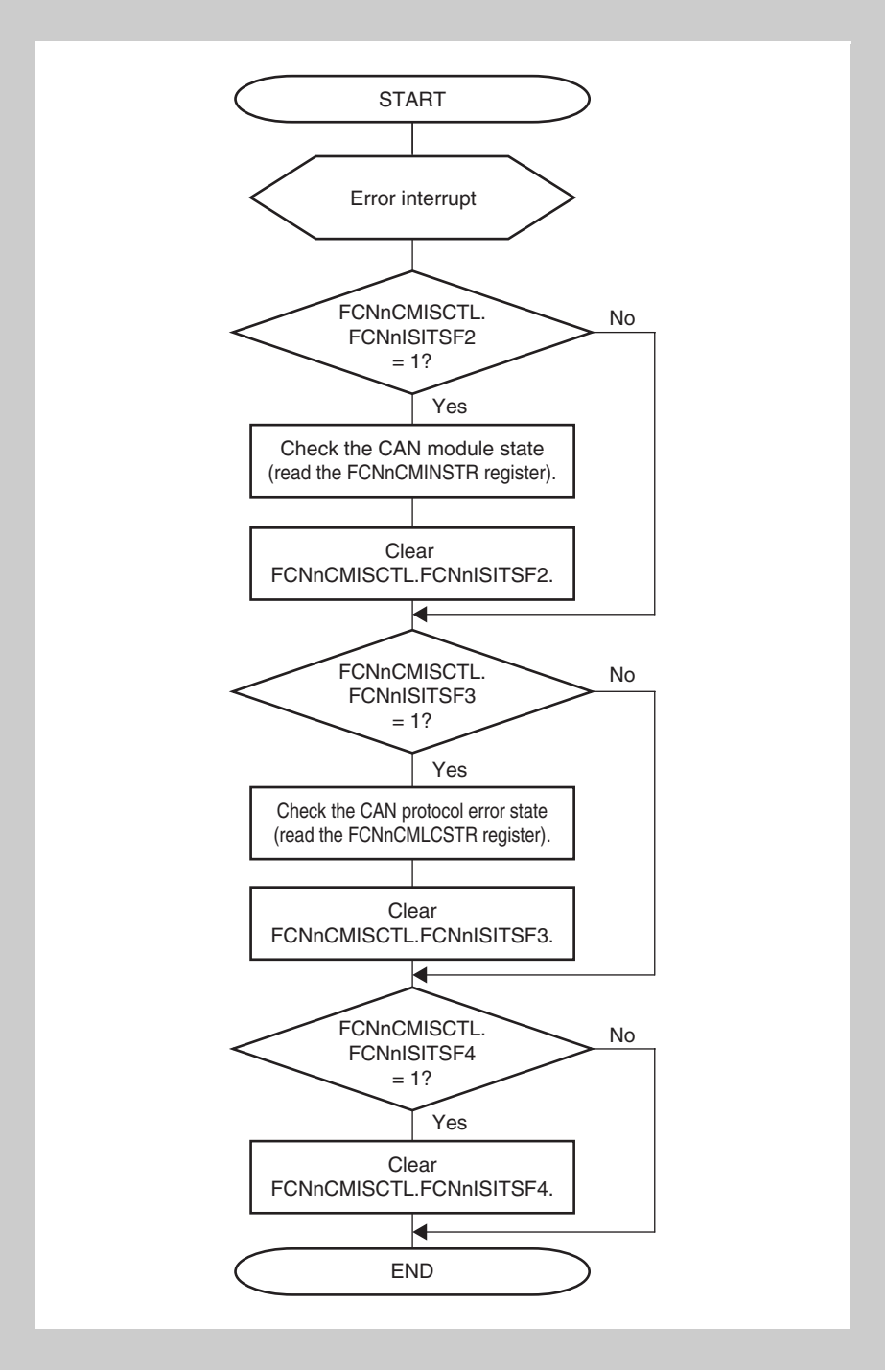


Figure 22-41 Error handling

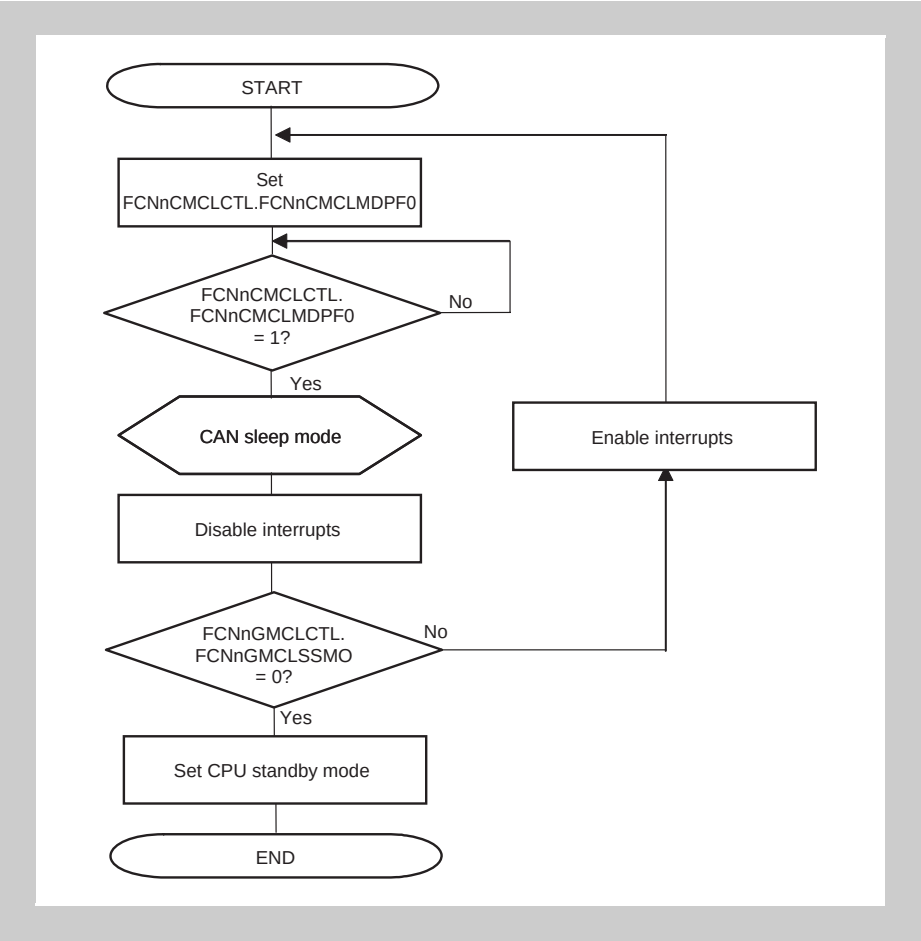


Figure 22-42 Setting CPU stand-by (from FCN sleep mode)

Notes

1. Before the CPU is set in the CPU standby mode, please check if the FCN sleep mode has been reached.
   However, after check of the FCN sleep mode, until the CPU is set in the CPU standby mode, the FCN sleep mode may be cancelled by wakeup from CAN bus.
2. There is a possibility, that between the check of FCNnGMCLSSMO = 0 and setting of the CPU standby mode a wake up condition on the CAN bus occurs. In that case the CAN module releases the SLEEP mode, the FCNnCMISITSF5 bit is set and if enabled the wake up interrupt will be generated.

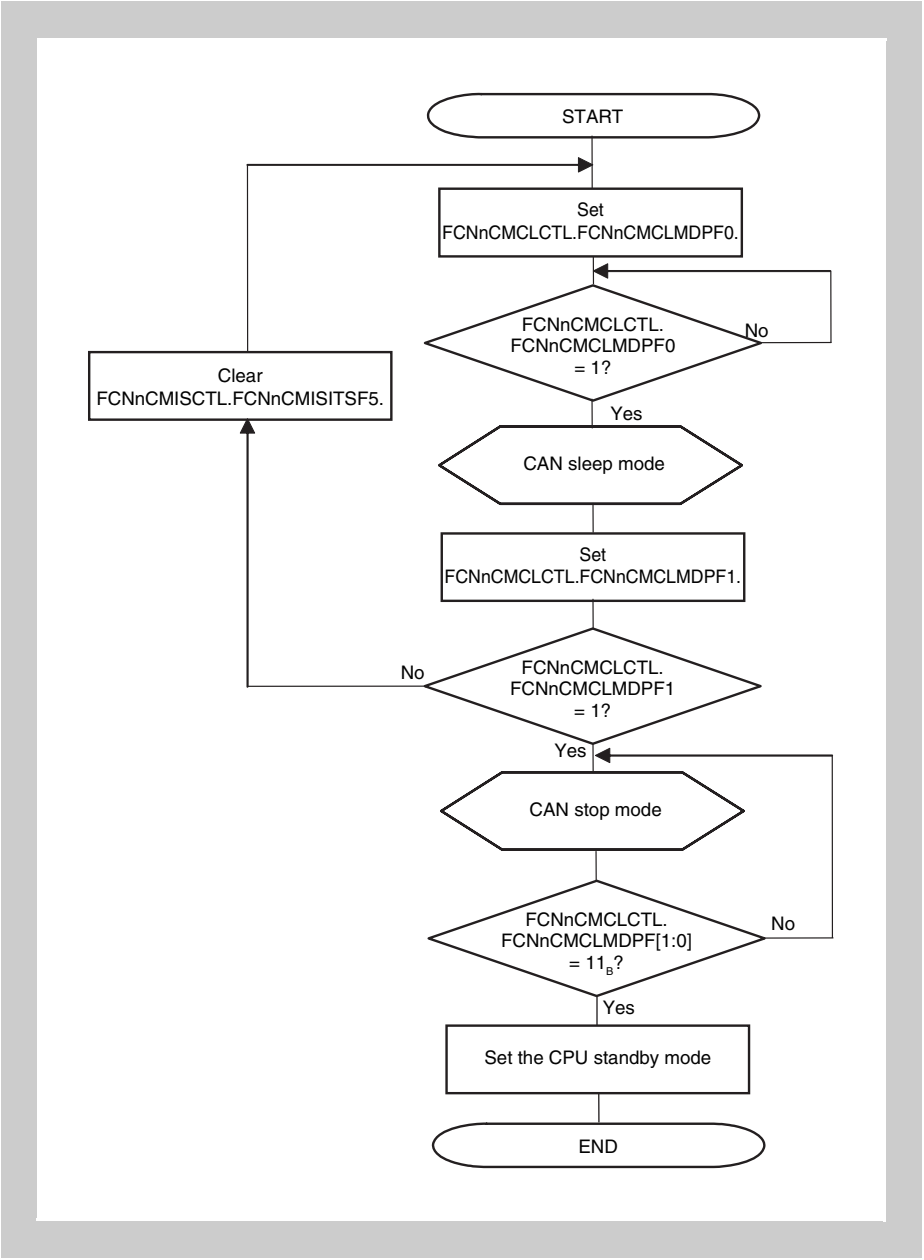


Figure 22-43 Setting CPU stand-by (from FCN stop mode)

Caution The FCN stop mode can only be released by setting FCNnCMCLCTL.FCNnCMCLMDPF[1:0] to 01_B and not by a change in the FCN bus state.

Chapter 23 Ethernet Controller (ETHA)

23.1 General

The Ethernet controller (ETHA) includes a 10/100 Mbps Ethernet Media Access Controller (MAC) conforming to IEEE802.3, a FIFO controller for flow control, and a checksum calculation unit (only for received packets) conforming to RFC1071.

23.1.1 V850E2/Sx4-H ETHA features

Instances

This microcontroller has following number of instances of ETHA:

Table 23-1 Instances of ETHA

ETHA	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of instances	0	0	1
Name	–	–	ETHA0

Instances index n

Throughout this chapter, the individual instances of ETHA is identified by the index "n" (n = 0), for example, ETHAnMACC1 for the MAC configuration register.

Register addresses

All ETHA register addresses are given as addresses offset from the individual base address <ETHAn_base_USER> or <ETHAnC_base>.
The base addresses <ETHAn_base> and <ETHAnC_base> of each ETHA are listed in the following table:

Table 23-2 Register base addresses <ETHAn_base_USER> and <ETHAnC_base>

Ethernet function	Base address	Address
MAC control register	ETHAn_base	F093 2000H
Statistics counter	ETHAn_base	$F093\ 2000_H$
FIFO controller control register	ETHAn_base	$F093\ 2000_H$
Ethernet controller dedicated DMAC control register	ETHAn_base	$F093\ 2000_H$
Transmission checksum dedicated DMAC control register	ETHAnC_base	$F093\ 3300_H$

Clock supply

The following clock is supplied to ETHA:

Table 23-3 ETHAn clock supply

ETHAn	Clock	Connected to:
ETHA0	HCLK	Clock generator CKSCLK_000/2

Interrupts and DMA

ETHA can generate the following interrupt requests and DMA requests:

Table 23-4 ETHAn interrupt requests and DMA requests

ETHAn signals	Function	Connected to:
ETHA0:		
INTETMRQ	Ethernet receive data ready interrupt	Interrupt controller INTETHA0SRX DMA controller trigger 101
INTETMRX	Ethernet packet reception interrupt	Interrupt controller INTETHA0SCRX DMA controller trigger 102
INTETMTX	Ethernet packet transmission interrupt	Interrupt controller INTETHA0SCTX DMA controller trigger 103
INTETMFS	Ethernet FIFO status interrupt	Interrupt controller INTETHA0FS DMA controller trigger 106
INTETMTS	Ethernet transmission status interrupt	Interrupt controller INTETHA0TS DMA controller trigger 105
INTETMRS	Ethernet reception status interrupt	Interrupt controller INTETHA0RS DMA controller trigger 104
INTETMOV	Ethernet MAC interrupt	Interrupt controller INTETHA0MAC
IRQSCTX_TCH	Ethernet transmit data calculation completion interrupt	Interrupt controller INTETHA0SCRXTCH
IRQSCRX_TCH	Ethernet transmit checksum interrupt	Interrupt controller INTETHA0SCTXTCH

Assignment of descriptor and data buffer

Descriptors and data buffers can be assigned to the following area:

- HBUS-RAM

ETHA hardware reset

ETHA and its registers are initialized by the following reset signal:

Table 23-5 ETHAn reset signal

ETHAn	Reset signal
ETHA0	System reset SYSRES

I/O signals The I/O signals of ETHA are listed in the following table:

Table 23-6 ETHAn I/O signals

ETHAn signal	Function	Connected to:
ETHA0:		
TXCLK (I)	Transmission clock	Port ETH0TXCLK
TXD[3:0] (O)	Transmission data	Port ETH0TXD[3:0]
TXEN (O)	Transmission data enable	Port ETH0TXEN
TXER (O)	Transmission error	Port ETH0TXER
COL (I)	Collision detection	Port ETH0COL
CRS (I)	Carrier detection	Port ETH0CRSDV
RXCLK (I)	Reception clock	Port ETH0RXCLK
RXD[3:0] (I)	Reception data	Port ETH0RXD[3:0]
RXDV (I)	Reception valid data	Port ETH0RXDV
RXER (I)	Reception error	Port ETH0RXER
MDC (O)	Serial transfer clock output	Port ETH0MDC
MDI (I)	Serial input	Port ETH0MDI
MDO (O)	Serial output	Port ETH0MDO
MDOEN (O)	–	–
FULLD (O)	–	–

23.1.2 Functions

(1) MAC

- 10/100 Mbps full-duplex communication, half-duplex communication, and flow control conforming to IEEE802.3 supported
- MII supported as physical layer device (PHY) interface
- Accessing PHY registers via serial management interface supported
- Statistics counter to support RMON/SNMP (RFC2665, RFC2819)
- Packet filtering based on address types
- VLAN frame detection

(2) FIFO

- Transmit/receive FIFO size: Transmit FIFO = 2 KB, receive FIFO = 2 KB
- FIFO status register
- Interrupts generated according to transmission/reception status and FIFO status

(3) DMAC for Ethernet controller

- Data transfer (DMA)
- Reception status DMA transfer
- Reading (in pointer chain format), analyzing, and writing back buffer descriptors
- Controlling interrupts in packet transfers

(4) Checksum calculation

- Transmit checksum calculation function conforming to RFC1071

  DMAC for transmit checksum can calculate multiple checksums successively and save the result to any address.
- Receive checksum calculation conforming to RFC1071

  The MAC header and FCS of a received packet are automatically identified and the checksum for verifying the received packet (excluding the dummy header) is generated.

23.2 Configuration

23.2.1 System configuration

The Ethernet controller transmits and receives data by using a dedicated direct memory access controller (DMAC). The Ethernet controller supports the MII (Media Independent Interface) of IEEE802.3 and can create a 10 Mbps or 100 Mbps Ethernet environment when it is connected to a PHY device conforming to MII. In addition, data can be communicated in full-duplex or half-duplex mode, which can be selected.

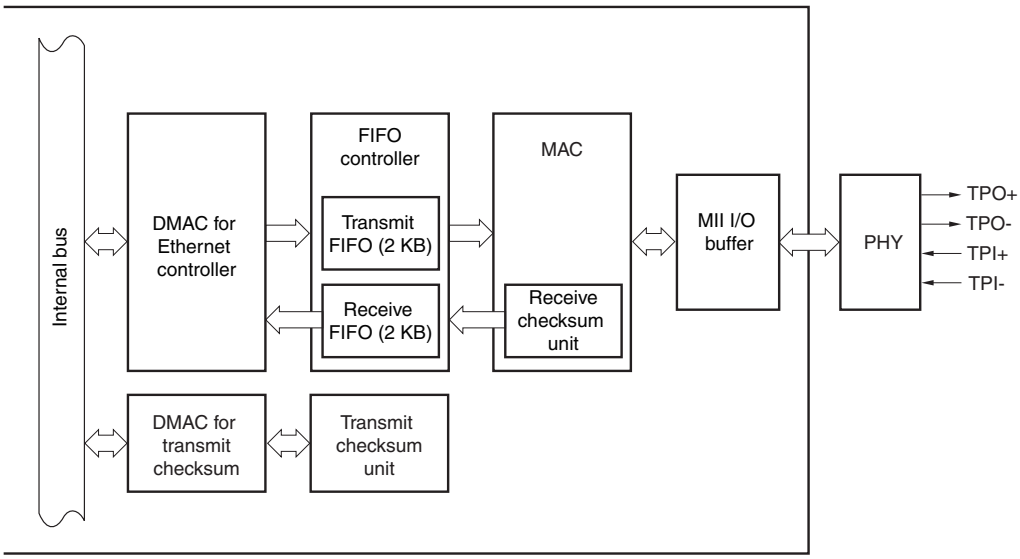


Figure 23-1 Configuration of Ethernet controller

(1) MAC

This unit has MAC functions and supports MII-based interfacing with an external PHY device.

- Receive checksum unit
  This unit calculates the receive checksum.

(2) FIFO controller

This unit controls the transmit/receive FIFO buffers.

2 KB FIFO buffers are separately available for transmission and reception.

(3) DMAC for Ethernet controller

This DMA controller controls data transmission and reception with the internal bus.

Caution The DMAC for the Ethernet controller processes all the data the Ethernet controller transmits and receives. Data cannot be transmitted or received in packet units by reading or writing a register.

(4) Transmit Checksum DMAC

This DMA controller that interfaces with the internal bus is dedicated to the transmit checksum function.

(5) Transmit checksum unit

This unit can only calculate the transmit checksum. Independent of the data transmission/reception interface, the transmit checksum unit calculates and transfers transmit data by using the transmit checksum DMA function and descriptor.

23.2.2 Interrupt requests and sources

The Ethernet controller interrupt requests and their sources are listed below.

Table 23-7 Interrupt requests

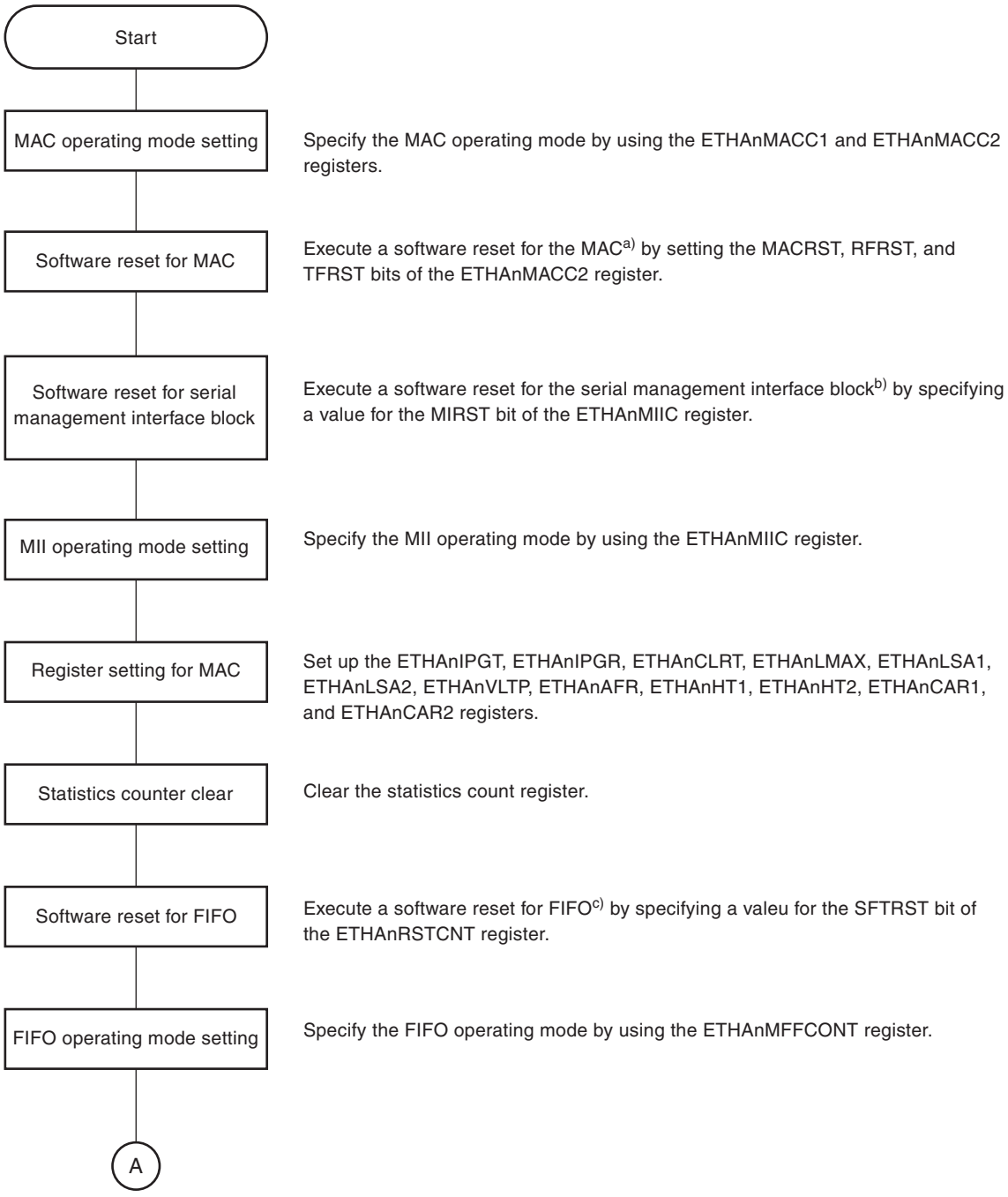
Interrupt request		Interrupt source
INTETMRQ	Ethernet receive data ready interrupt	Received packet read request
INTETMRX	Ethernet packet reception interrupt	Packet reception (DMA) completion interrupt (RXI)
		Reception (DMA) end of chain interrupt (RECI)
		Receive data buffer access error interrupt (RBEI)
		Pause interrupt (RUPI) triggered by the “U” (used) bit in the receive descriptor
INTETMTX	Ethernet packet transmission interrupt	Packet transmission (DMA) completion interrupt (TXI)
		Transmission (DMA) end of chain interrupt (TECI)
		Transmit data buffer access error interrupt (TBEI)
		Pause interrupt (TUPI) triggered by the “U” (used) bit in the transmit descriptor
INTETMFS	Ethernet FIFO status interrupt	FIFO status (ETHAnFSTATUS) interrupt
INTETMTS	Ethernet transmission status interrupt	Transmission status (ETHAnFTXSTATUS) interrupt
INTETMRS	Ethernet reception status interrupt	Reception status (ETHAnRXSTATUS) interrupt
INTETMOV	Ethernet MAC interrupt	Statistics counter overflow (CARRY status)
IRQSCTX_TCH	Ethernet transmit data calculation completion interrupt	1 transmit checksum calculation completion interrupt (TCH_TXI)
		All transmit checksums calculation completion interrupt (TCH_TECI)
		Transmit checksum buffer access error interrupt (TCH_TBEI)
		Transmit checksum calculation pause interrupt (TCH_RUPI)
IRQSCRX_TCH	Ethernet transmit checksum interrupt	1 transmit checksum writing completion interrupt (TCH_RXI)
		All transmit checksums calculation writing completion interrupt (TCH_RECI)
		Data write error interrupt (TCH_RBEI)

- Each interrupt source can be masked. If an interrupt source is generated while the interrupt is masked, the corresponding bit in the status register is set but the interrupt request is not generated.
- It is recommended to read the interrupt register if multiple sources have been generated concurrently.

23.3 Initialization

Use the following procedure to perform initialization before using the V850E2/Sx4-H Ethernet controller:

1. Enable Ethernet controller operation.
2. Initialize the media access controller (MAC).
3. Initialize the FIFO controller.
4. Initialize the DMA controller.
5. Set up interrupts.

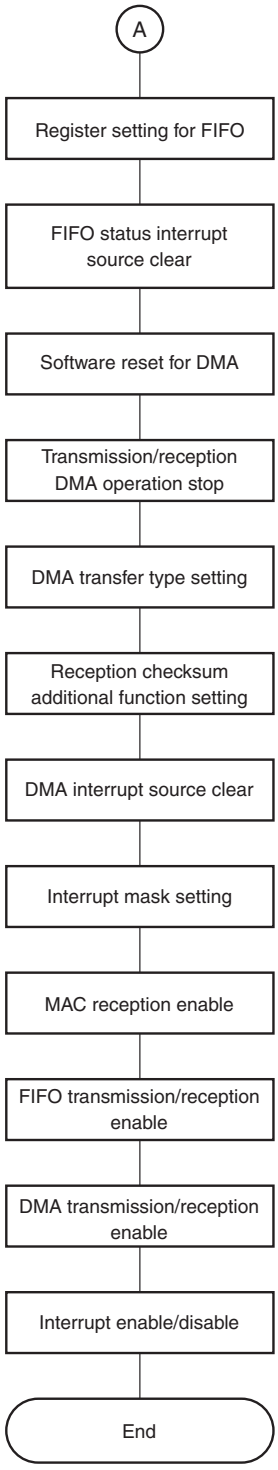


a) To cancel the software reset for the MAC, set the MCRST, RFRST, and TFRST bits of the ETHAnMACC2 register at the same time, and then clear them at the same time. Manipulate these bits at an interval of five or more RXCLK or TXCLK clock cycles.

b) To cancel the software reset for the serial management interface block, set the MIRST bit of the ETHAnMIIC register, and then clear it. Manipulate this bit at an interval of five or more Ethernet control clock cycles (f_{EC}).

c) The SFTRST bit of the ETHAnRSTCNT register is a self-clear bit that is automatically cleared after it is set.

Figure 23-2 Initializing Ethernet controller (1/2)



Set up the ETHAnFLOWTH, ETHAnPAUSETM, ETHAnRXERSEL, ETHAnTXABTCNT, and ETHAnRXABTCNT registers.

Clear the ETHAnFSTATUS, ETHAnTXSTATUS, and ETHAnRXSTATUS registers[a].

Execute a software reset for the DMA controller by specifying a value for the SFTRST bit of the ETHAnSFTRST register[b].

Stop the transmission/reception DMA operation by specifying a value for the TXEN and RXEN bits of the ETHAnTRANSCTL register.
The DMA operating status applied to the TXEN_STA and RXEN_STA bits of the ETHAnTRANSCTL register.

Specify the DMA transfer type for the DMA controller by using the ETHAnDMACM register.

Enable or disable the reception checksum function by specifying a value for the RXCHKSMEN bit of the ETHAnTRANSCTL register.

Clear the ETHAnINTMS register[c].

Specify the interrupt mask settings by using the ETHAnFSTATMK, ETHAnTXSTATMK, ETHAnRXSTATMK, and ETHAnINTMS registers.

Enable the reception data interface function for the MAC by setting the SRXEN bit of the ETHAnMACC1 register.

Enable the transmission/reception function for FIFO by setting the RXEN and TXEN bits of the ETHAnMFFCONT register.

Enable the transmission/reception DMA transfer function for the DMA controller by setting the RXEN and TXEN bits of the ETHAnTRANSCTL register.

Set up the interrupt controller control registers.

a) The ETHAnFSTATUS, ETHAnTXSTATUS, and ETHAnRXSTATUS are read-clear registers.
b) The SFTRST bit of the ETHAnSFTRST register is a self-clear bit that is automatically cleared after it is set.
c) The source bits of the ETHAnINTMS register RBEI, RECI, RXI, TBEI, TECI, and TXI are read-clear bits.

Figure 23-3 Initializing Ethernet controller (2/2)

23.4 Registers for Controlling the Ethernet Controller

(1) Register setting procedure

To update the values of the control registers, make sure that transmission and reception of frames and DMA are stopped.

If these registers are updated while frames are being transmitted or received, or while DMA is in progress, the operation is not guaranteed.

Caution When using the Ethernet controller, the FIFO controller control register (ETHAnMFFCONT) must be set up first. For details, see *(1) "ETHAnMFFCONT - FIFO controller control register" on page 1686 in 23.4.3 "FIFO controller control registers"*.

(2) Register list

Table 23-8 MAC control register list

	Offset address	Symbol	Register name	R/W	Access bit unit			Default value
					8	16	32	
	0000_H	ETHAnMACC1	MAC setting register 1	R/W			✓	00000000_H
	0004_H	ETHAnMACC2	MAC setting register 2	R/W			✓	00000000_H
	0008_H	ETHAnIPGT	Back-to-back IPG register	R/W			✓	00000013_H
	$000C_H$	ETHAnIPGR	Non back-to-back IPG register	R/W			✓	$00000E13_H$
	0010_H	ETHAnCLRT	Collision register	R/W			✓	$0000380F_H$
	0014_H	ETHAnLMAX	Maximum packet length register	R/W			✓	00000600_H
	0054_H	ETHAnLSA1	Station address register 1	R/W			✓	00000000_H
	0058_H	ETHAnLSA2	Station address register 2	R/W			✓	00000000_H
	$005C_H$	ETHAnPTVR	Pause timer value read register	R			✓	00000000_H
	0064_H	ETHAnVLTP	VLAN type register	R/W			✓	00000000_H
	0080_H	ETHAnMIIC	MII configuration register	R/W			✓	00000000_H
	0094_H	ETHAnMCMD	MII command register	W			✓	00000000_H
	0098_H	ETHAnMADR	MII address register	R/W			✓	00000000_H
	$009C_H$	ETHAnMWTD	MII write data register	R/W			✓	00000000_H
<R>	$00A0_H$	ETHAnMRDD	MII read data register	R			✓	00000000_H
<R>	$00A4_H$	ETHAnMIND	MII indicator register	R			✓	00000000_H
	$00C8_H$	ETHAnAFR	Address filter register	R/W			✓	00000000_H
	$00CC_H$	ETHAnHT1	Hash table register 1	R/W			✓	00000000_H
	$00D0_H$	ETHAnHT2	Hash table register 2	R/W			✓	00000000_H
	$00DC_H$	ETHAnCAR1	Carry register 1	R/W			✓	00000000_H
	$00E0_H$	ETHAnCAR2	Carry register 2	R/W			✓	00000000_H
	0130_H	ETHAnCAM1	Carry mask register 1	R/W			✓	00000000_H
	0134_H	ETHAnCAM2	Carry mask register 2	R/W			✓	00000000_H

Table 23-9 Statistics counter register list

Offset address	Symbol	Register name	R/W	Access bit unit			Default value
				8	16	32	
0140_H	ETHAnRBYT	Reception byte counter	R/W			✓	00000000_H
0144_H	ETHAnRPKT	Reception packet counter	R/W			✓	00000000_H
0148_H	ETHAnRFCS	Reception FCS error frame counter	R/W			✓	00000000_H
$014C_H$	ETHAnRMCA	Reception multicast packet counter	R/W			✓	00000000_H
0150_H	ETHAnRBCA	Reception broadcast packet counter	R/W			✓	00000000_H
0154_H	ETHAnRXCF	Reception control frame packet counter	R/W			✓	00000000_H
0158_H	ETHAnRXPF	Reception pause frame packet counter	R/W			✓	00000000_H
$015C_H$	ETHAnRXUO	Reception undefined control packet counter	R/W			✓	00000000_H
0160_H	ETHAnRALN	Reception alignment error counter	R/W			✓	00000000_H
0164_H	ETHAnRFLR	Reception frame length error counter	R/W			✓	00000000_H
0168_H	ETHAnRCDE	Reception code error counter	R/W			✓	00000000_H
$016C_H$	ETHAnRFCR	Reception false carrier counter	R/W			✓	00000000_H
0170_H	ETHAnRUND	Reception undersize packet counter	R/W			✓	00000000_H
0174_H	ETHAnROVR	Reception oversize packet counter	R/W			✓	00000000_H
0178_H	ETHAnRFRG	Reception fragment counter	R/W			✓	00000000_H
$017C_H$	ETHAnRJBR	Reception jabber counter	R/W			✓	00000000_H
0180_H	ETHAnR64	Receive 64-byte frame counter	R/W			✓	00000000_H
0184_H	ETHAnR127	Receive 65- to 127-byte frame counter	R/W			✓	00000000_H
0188_H	ETHAnR255	Receive 128- to 255-byte frame counter	R/W			✓	00000000_H
$018C_H$	ETHAnR511	Receive 256- to 511-byte frame counter	R/W			✓	00000000_H
0190_H	ETHAnR1K	Receive 512- to 1023-byte frame counter	R/W			✓	00000000_H
0194_H	ETHAnRMAX	Receive 1024- to RMAX-byte frame counter	R/W			✓	00000000_H
0198_H	ETHAnRVBT	Receive valid byte counter	R/W			✓	00000000_H
$01C0_H$	ETHAnTBYT	Transmission byte counter	R/W			✓	00000000_H
$01C4_H$	ETHAnTPKT	Transmission packet counter	R/W			✓	00000000_H
$01C8_H$	ETHAnTFCS	Transmission FCS error frame counter	R/W			✓	00000000_H
$01CC_H$	ETHAnTMCA	Transmission multicast packet counter	R/W			✓	00000000_H
$01D0_H$	ETHAnTBCA	Transmission broadcast packet counter	R/W			✓	00000000_H
$01D4_H$	ETHAnTUCA	Transmission unicast packet counter	R/W			✓	00000000_H
$01D8_H$	ETHAnTXPF	Transmission pause control frame counter	R/W			✓	00000000_H
$01DC_H$	ETHAnTDFR	Transmission delay packet counter	R/W			✓	00000000_H
$01E0_H$	ETHAnTXDF	Transmission excessive delay packet counter	R/W			✓	00000000_H
$01E4_H$	ETHAnTSCL	Transmission single collision packet counter	R/W			✓	00000000_H
$01E8_H$	ETHAnTMCL	Transmission multiple collision packet counter	R/W			✓	00000000_H
$01EC_H$	ETHAnTLCL	Transmission late collision packet counter	R/W			✓	00000000_H
$01F0_H$	ETHAnTXCL	Transmission excessive collision packet counter	R/W			✓	00000000_H
$01F4_H$	ETHAnTNCL	Transmission total collision counter	R/W			✓	00000000_H
$01F8_H$	ETHAnTCSE	Transmission carrier sense error counter	R/W			✓	00000000_H
$01FC_H$	ETHAnTIME	MAC internal error counter	R/W			✓	00000000_H

Table 23-10 FIFO controller register list

Offset address	Symbol	Register name	R/W	Access bit unit			Default Value
				8	16	32	
0200_H	ETHAnMFFCONT	FIFO controller control register	R/W			✓	00000000_H
0204_H	ETHAnRSTCNT	Software reset control register	R/W			✓	00000000_H
0218_H	ETHA0FLOWTH	Flow control threshold value register	R/W			✓	06000200_H
$021C_H$	ETHAnPAUSETM	Pause timer value register	R/W			✓	$7FFFFFFF_H$
0220_H	ETHAnRXERSEL	Receive error selection register	R/W			✓	00000001_H
0230_H	ETHAnTXSTMONI1	Transmission status monitor 1 register	R			✓	00000000_H
0234_H	ETHAnTXSTMONI2	Transmission status monitor 2 register	R			✓	00000000_H
0238_H	ETHAnTXFINF1	Transmission status 1 register	R			✓	00000800_H
$023C_H$	ETHAnTXFINF2	Transmission status 2 register	R			✓	00000001_H
0240_H	ETHAnRXSTMONI	Reception status monitor register	R			✓	00000000_H
0244_H	ETHAnRXFINF1	Reception status 1 register	R			✓	00000000_H
0248_H	ETHAnRXFINF2	Reception status 2 register	R			✓	00000800_H
$024C_H$	ETHAnRXFINF3	Reception status 3 register	R			✓	00000001_H
0250_H	ETHAnFSTATUS	FIFO status interrupt register	R			✓	00000000_H
0254_H	ETHA0FSTATMK	FIFO status interrupt mask register	R/W			✓	$01011FFF_H$
0258_H	ETHAnTXSTATUS	Transmission status interrupt register	R			✓	00000000_H
$025C_H$	ETHAnTXSTATMK	Transmission status interrupt mask register	R/W			✓	$000101FF_H$
0260_H	ETHAnRXSTATUS	Reception status interrupt register	R			✓	00000000_H
0264_H	ETHAnRXSTATMK	Reception status interrupt mask register	R			✓	$00007FFF_H$
0270_H	ETHAnTXABTCNT	TX abort counter	R/W			✓	00000000_H
0274_H	ETHAnRXABTCNT	RX abort counter	R			✓	00000000_H

Table 23-11 DMAC for Ethernet controller register list

Offset address	Symbol	Register name	R/W	Access bit unit			Default value
				8	16	32	
0300_H	ETHAnMODE	Core function setting register	R/W			✓	00000000_H
0304_H	ETHAnINTMS	Interrupt control register	R/W			✓	07000700_H
0308_H	ETHAnTRANSCTL	Transfer control register	R/W			✓	00030000_H
$030C_H$	ETHAnSFTRST	Software reset control register	R/W			✓	00000000_H
0310_H	ETHAnDMACM	DMA mode control register	R/W			✓	00000010_H
0320_H	ETHAnRXDP	Receive descriptor pointer register	R/W			✓	$FFFFFFFC_H$
0324_H	ETHAnLSTRXDP	Last receive descriptor pointer register	R			✓	$FFFFFFFC_H$
0328_H	ETHAnTXDP	Transmit descriptor pointer register	R/W			✓	$FFFFFFFC_H$
$032C_H$	ETHAnLSTTXDP	Last transmit descriptor pointer register	R			✓	$FFFFFFFC_H$

Table 23-12 DMAC control for transmit checksum register list

Offset address	Symbol	Register name	R/W	Access bit unit			Default value
				8	16	32	
0300_H	ETHAnCMODE	Transmit checksum unit function setting register	R/W			✓	00000000_H
0304_H	ETHAnCINTMS	Transmit checksum interrupt register	R/W			✓	07000700_H
0308_H	ETHAnCTRANSCTL	Transmit checksum transfer control register	R/W			✓	00030000_H
$030C_H$	ETHAnCSFTRST	Transmit checksum software reset register	R/W			✓	00000000_H
0310_H	ETHAnCDMACM	Transmit checksum DMA control mode setting register	R/W			✓	00000010_H
0320_H	ETHAnCRXDP	Transmit checksum receive descriptor pointer register	R/W			✓	$FFFFFFFC_H$
0324_H	ETHAnCLSTRXDP	Transmit checksum last receive descriptor pointer register	R/W			✓	$FFFFFFFC_H$
0328_H	ETHAnCTXDP	Transmit checksum transmit descriptor pointer register	R/W			✓	$FFFFFFFC_H$
$032C_H$	ETHAnCLSTTXDP	Transmit checksum last transmit descriptor pointer register	R/W			✓	$FFFFFFFC_H$

23.4.1 MAC control registers

(1) ETHAnMACC1 - MAC setting register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0000_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. Be sure to execute a software reset after setting the operation mode. To execute a software reset, set the MCRST, RFRST, and TFRST bits of the ETHAnMACC2 register at the same time. Cancel the software reset by clearing these bits at the same time.
2. Be sure to set bits 31 to 15, 13, 12, and 4 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	MACLB	0	0	TXFC	RXFC	SRXEN	PARF
R	R/W	R	R	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
PUREP	FLCHT	NOBO	0	CRCEN	PADEN	FULLD	HUGEN
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W

Table 23-13 ETHAnMACC1 register contents (1/2)

Bit position	Bit name	Function
14	MACLB	MAC loopback 0: Disables loopback operation. 1: Operation loops back from transmission block to reception block in MAC. To execute the loopback operation, set the FULLD bit to enable full-duplex operation.
11	TXFC	Transmission flow control enable 0: Disables transmission of a pause control frame executed by inputting the TPCF signal. 1: Enables transmission of a pause control frame executed by inputting the TPCF signal.
10	RXFC	Reception flow control enable 0: A pause operation is not executed. 1: A pause operation is executed for the pause period set to the pause timer. The value of the pause timer is updated, regardless of the setting of this bit, when a valid pause control frame is received.

Table 23-13 ETHAnMACC1 register contents (2/2)

Bit position	Bit name	Function
9	SRXEN	Reception enable 0: Reception is disabled. 1: The function of the reception data interface is enabled. If the setting of this bit is changed while the CRS signal is asserted, the new setting becomes valid after the CRS signal has been deasserted, regardless of the setting of the FULLD bit.
8	PARF	Control packet pass 0: A control frame is judged as a control frame. 1: No received packet, including a control frame, is judged as a control frame. The value of the pause timer is not updated even if a valid pause control frame is received, regardless of the setting of the RXFC bit.
7	PUREP	Pure preamble 0: The data of a preamble is not checked. 1: A reception status interrupt is generated if an illegal preamble is detected.
6	FLCHT	Length field check 0: The length field is not checked. 1: The value of the length field and data field length are checked, and a status interrupt is generated.
5	NOBO	No backoff 0: Packets are transmitted by using the backoff algorithm. 1: Packets are always transmitted without using the backoff algorithm.
3	CRCEN	CRC appending 0: CRC is not appended. The end of the transmitted packet must be a valid frame check sequence (FCS). The MAC checks the FCS, and, if the FCS value is not correct, the MAC generates a transmission status interrupt (ETHAnTXSTATUS) to report an error. 1: CRC is automatically appended to the end of a packet. An internally generated frame check sequence (FCS) is appended to the end of the transmit packet.
2	PADEN	PAD appending 0: PAD is not appended. 1: PAD is appended to packets that are less than 64 bytes long. At this time, CRC is automatically appended to the end of the packet regardless of the setting of the CRCEN bit.
1	FULLD	Full-duplex enable 0: Half-duplex operation 1: Full-duplex operation
0	HUGEN	Huge packet enable 0: Transmission/reception of a packet that exceeds the value of the maximum packet length register (ETHAnLMAX) is stopped. 1: Transmission/reception of a packet that exceeds the value of the maximum packet length register (ETHAnLMAX) is not stopped.

(2) ETHAnMACC2 - MAC setting register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0004_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. Be sure to execute a software reset after setting the operation mode. To execute a software reset, set the MCRST, RFRST, and TFRST bits of the ETHAnMACC2 register at the same time. Cancel the software reset by clearing these bits at the same time.

   Manipulate these reset bits at an interval of five or more RXCLK or TXCLK cycles.

2. Be sure to set bits 31 to 11, 7, and 3 to 0 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	MCRST	RFRST	TFRST
R	R	R	R	R	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	BPNB	APD	VPD	0	0	0	0
R	R/W	R/W	R/W	R	R	R	R

Table 23-14 ETHAnMACC2 register contents

Bit position	Bit name	Function
10	MCRST	MAC control block software reset 0: Cancels a software reset for the MAC control block. 1: Executes a software reset for the MAC control block.
9	RFRST	Reception block software reset 0: Cancels a software reset for the reception block. 1: Executes a software reset for the reception block.
8	TFRST	Transmission block software reset 0: Cancels a software reset for the transmission block. 1: Executes a software reset for the transmission block.
6	BPNB	No backoff after back pressure When this bit is set, backoff is not performed for a transmission after back pressure.
5	APD	Auto VLAN PAD If a packet that matches the VLAN type registered to the ETHAnVLTP register is transmitted, it is treated as a VLAN packet and PAD is appended.
4	VPD	VLAN pad mode The packet to be transmitted is always treated as a VLAN packet and PAD is appended.

(3) ETHAnIPGT - Back-to-back IPG register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0008_H

Initial value $0000\ 0013_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 7 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	IPGT						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-15 ETHAnIPGT register contents

Bit position	Bit name	Function
6 to 0	IPGT	IPG in back-to-back transmission These bits specify the gap between packets (inter-packet gap (IPG)) in back-to-back transmission. The expression used to calculate the IPG is as follows. • IPG = (5 + IPGT) x time required to transmit 4 bits (Time required to transmit 1 bit = 100 ns when the data rate is 10 Mbps or 10 ns when the data rate is 100 Mbps) Set the IPG to the time required to transmit at least 96 bits to satisfy the specification of IEEE802.3 (see *(5) “Inter-packet gap (IPG)” on page 1740 in 23.5.2 “Frame transmission”*).

(4) ETHAnIPGR - Non back-to-back IPG register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $000C_H$

Initial value $0000\ 0E13_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 15 and 7 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	IPGR1						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	IPGR2						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-16 ETHAnIPGR register contents

Bit position	Bit name	Function
14 to 8	IPGR1	Carrier sense period These bits specify the carrier sense period of the first half of the IPG in transmission other than back-to-back transmission. The calculation expression used to calculate the carrier sense period is as follows. • Carrier sense period = (2 + IPGR1) x time required to transmit 4 bits Set the carrier sense period to 2/3IPG to satisfy the specification of IEEE802.3 (see *(5) “Inter-packet gap (IPG)” on page 1740 in 23.5.2 “Frame transmission”*).
6 to 0	IPGR2	IPG in transmission other than back-to-back transmission These bits specify the IPG in transmission other than back-to-back transmission. The expression used to calculate the IPG is as follows. • IPG = (5 + IPGR2) x time required to transmit 4 bits The carrier sense period specified by the IPGR1 bits is included in the IPG specified by the IPGR2 bits. Set the IPG to the time required to transmit at least 96 bits to satisfy the specification of IEEE802.3 (see *(5) “Inter-packet gap (IPG)” on page 1740 in 23.5.2 “Frame transmission”*).

(5) ETHAnCLRT - Collision register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0010_H

Initial value $0000\ 380F_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 14 and 7 to 4 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	LCOL					
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	RETRY			
R	R	R	R	R/W	R/W	R/W	R/W

Table 23-17 ETHAnCLRT register contents

Bit position	Bit name	Function
13 to 8	LCOL	Collision window These bits specify the collision window width. The width of the collision window to be set is calculated by the following expression. • Collision window width = (LCOL + 8) x time required to transmit 8 bits The IEEE802.3 defines the collision window width as the time required to transmit 512 bits.
3 to 0	RETRY	Maximum number of times of retransmission in case of collision These bits specify the maximum number of times to attempt retransmission when a collision occurs. If retransmission does not finish within the value specified by these bits, transmission is aborted. This value indicates the maximum number of collisions. The IEEE802.3 defines the maximum number of collisions as 15.

(6) ETHAnLMAX - Maximum packet length register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0014_H

Initial value $0000\ 0600_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
MAXF15	MAXF14	MAXF13	MAXF12	MAXF11	MAXF10	MAXF9	MAXF8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
MAXF7	MAXF6	MAXF5	MAXF4	MAXF3	MAXF2	MAXF1	MAXF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-18 ETHAnLMAX register contents

Bit position	Bit name	Function
15 to 0	MAXF[15:0]	Maximum packet length (bytes) When the ETHAnMACC1.HUGEN bit is 0, the transmit and receive packet length is limited by the value specified by these bits. During reception: Reception is terminated immediately when the receive frame length exceeds the value specified by the MAXF bits. During transmission: Transmission is aborted immediately when the transmit frame length exceeds the value specified by the MAXF bits.

(7) ETHAnLSA1 - Station address register 1

This register is used to compare a source address when a pause control frame is assembled and a destination address when address filtering is used. This register is used in combination with the ETHAnLSA2 register as a 48-bit register.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0054_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
LSA115	LSA114	LSA113	LSA112	LSA111	LSA110	LSA19	LSA18
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
LSA17	LSA16	LSA15	LSA14	LSA13	LSA12	LSA11	LSA10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-19 ETHAnLSA1 register contents

Bit position	Bit name	Function
15 to 0	LSA1[15:0]	Station address (SA) (47:32) The SA bits (47:0) are used to compare a source address when a pause control frame is assembled and a destination address when address filtering is used (see *(a) “Filtering of unicast addresses” on page 1756 in (1) "Overview of address filtering" of 23.5.7 “Address filtering”*).

(8) ETHAnLSA2 - Station address register 2

This register is used to compare a source address when a pause control frame is assembled and a destination address when address filtering is used. This register is used in combination with the ETHAnLSA1 register as a 48-bit register.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0058_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
LSA231	LSA230	LSA229	LSA228	LSA227	LSA226	LSA225	LSA224
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
LSA223	LSA222	LSA221	LSA220	LSA219	LSA218	LSA21	LSA216
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
LSA215	LSA214	LSA213	LSA212	LSA211	LSA210	LSA29	LSA28
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
LSA27	LSA26	LSA25	LSA24	LSA23	LSA22	LSA21	LSA20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-20 ETHAnLSA2 register contents

Bit position	Bit name	Function
31 to 0	LSA2[31:0]	Station address (SA) (31:0) The SA bits (47:0) are used to compare a source address when a pause control frame is assembled and a destination address when address filtering is used (see *(a) “Filtering of unicast addresses” on page 1756 in (1) "Overview of address filtering" of 23.5.7 “Address filtering”*).

(9) ETHAnPTVR - Pause timer value read register

This register is used to read the value of the pause timer counter.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + $005C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
PTCT15	PTCT14	PTCT13	PTCT12	PTCT11	PTCT10	PTCT9	PTCT8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
PTCT7	PTCT6	PTCT5	PTCT4	PTCT3	PTCT2	PTCT1	PTCT0
R	R	R	R	R	R	R	R

Table 23-21 ETHAnPTVR register contents

Bit position	Bit name	Function
15 to 0	PTCT[15:0]	Pause timer counter These bits indicate the value currently set in the pause timer. The value of this register is valid only when reception flow control is enabled (the ETHAnMACC1.RXFC bit is 1) (see *(1) "Flow control" on page 1746 in 23.5.4 "MAC control function"*).

(10) ETHAnVLTP - VLAN type register

This register is used to specify the operation to be performed on a VLAN frame.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0064_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
VLTP15	VLTP14	VLTP13	VLTP12	VLTP11	VLTP10	VLTP9	VLTP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
VLTP7	VLTP6	VLTP5	VLTP4	VLTP3	VLTP2	VLTP1	VLTP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-22 ETHAnVLTP register contents

Bit position	Bit name	Function
15 to 0	VLTP[15:0]	VLAN frame operation These bits specify the operation to be performed or a VLAN frame (see *(3) “Operations related to VLAN frame” on page 1749 in 23.5.4 “MAC control function”*). During reception: The value of VLTP[15:0] is compared with the value of the TPID field (2 bytes following the source address) of a frame to detect a VLAN frame. During transmission: If the value of the VLAN field matches the value of VLTP[15:0] when the ETHAnMACC2.APD bit is 1, PAD is appended to the VLAN frame.

(11) ETHAnMIIC - Serial management interface configuration register

This register is used to set the operation mode of the serial management interface block.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0080_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. Manipulate the MIRST bit at an interval of five or more Ethernet controller clock cycles (f_{EC}).
2. Be sure to set bits 31 to 16, 14 to 5, and 0 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
MIRST	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	CLKS2	CLKS1	CLKS0	PHYSEL	0
R	R	R	R/W	R/W	R/W	R/W	R

Table 23-23 ETHAnMIIC register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15</td><td>MIRST</td><td>Serial management interface block software reset
0: Cancels a software reset for the serial management interface block.
1: Executes a software reset for the serial management interface block.</td></tr>
<tr><td>4 to 2</td><td>CLKS[2:0]</td><td>MDC division ratio
These bits select a division ratio according to an Ethernet controller clock (f_{EC}) to be used (see (a) “MDC clock” on page 1752 in (1) "Overview of serial management interface" of 23.5.6 “Serial management interface”).
To satisfy the specification of IEEE802.3, set a division ratio so that the MDC frequency is 2.5 MHz or less.
<table>
<tr><th>CLKS2</th><th>CLKS1</th><th>CLKS0</th><th>Input frequency of f_{EC}</th></tr>
<tr><td>0</td><td>0</td><td>1</td><td>33 MHz or less (division ratio: 14)</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>50 MHz or less (division ratio: 20)</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>66 MHz or less (division ratio: 28)</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>100 MHz or less (division ratio: 40)</td></tr>
<tr><td colspan="3">Other than above</td><td>Setting prohibited</td></tr>
</table>
</td></tr>
<tr><td>1</td><td>PHYSEL</td><td>MDC output setting
Set this bit if data is not correctly transferred during communication with PHY when the MDC is stopped.
1: The MDC is always output for any frames other than the management frame.
0: The MDC is stopped for frames other than the management frame.</td></tr>
</table>

(12) ETHAnMCMD - MII command register

This register is used to read an external PHY device by using the SCAN command and MII management interface.

Access This register is write-only, in 32-bit units.

The value written to the ETHAnMCMD register must be read from the ETHAnMIND register.

Address <ETHAn_base> + 0094_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 2 to “0”. Bits 1 and 0 can only be written.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	SCANC	RSTAT
R	R	R	R	R	R	W	W

Table 23-24 ETHAnMCMD register contents

Bit position	Bit name	Function
1	SCANC	SCAN command When this bit is set, the SCAN command is executed.
0	RSTAT	MII management read When this bit is set, the MII management interface reads the external PHY device.

(13) ETHAnMADR - MII address register

This register is used to set a PHY address and a PHY register address.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0098_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 13 and 7 to 5 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	FIAD				
R	R	R	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	RGAD				
R	R	R	R/W	R/W	R/W	R/W	R/W

Table 23-25 ETHAnMADR register contents

Bit position	Bit name	Function
12 to 8	FIAD	PHY address These bits specify a PHY address. One Ethernet controller can control up to 31 PHY devices.
4 to 0	RGAD	PHY register address These bits specify the address of the register to be accessed. The Ethernet controller can access 32 16-bit registers in one PHY device.

(14) ETHAnMWTD - MII write data register

This register is used to set the data to be written to an external PHY device when the MII management interface writes a PHY device.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $009C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
CTLD15	CTLD14	CTLD13	CTLD12	CTLD11	CTLD10	CTLD9	CTLD8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
CTLD7	CTLD6	CTLD5	CTLD4	CTLD3	CTLD2	CTLD1	CTLD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-26 ETHAnMWTD register contents

Bit position	Bit name	Function
15 to 0	CTLD[15:0]	MII write data This is a write data field when the MII management interface writes an external PHY device.

(15) ETHAnMRDD - MII read data register

This register is used to read data that has been read from an external PHY device by the MII management interface.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + $00A0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
PRSD15	PRSD14	PRSD13	PRSD12	PRSD11	PRSD10	PRSD9	PRSD8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
PRSD7	PRSD6	PRSD5	PRSD4	PRSD3	PRSD2	PRSD1	PRSD0
R	R	R	R	R	R	R	R

Table 23-27 ETHAnMRDD register contents

Bit position	Bit name	Function
15 to 0	PRSD[15:0]	MII read data This is a read data field when the MII management interface reads an external PHY device.

(16) ETHAnMIND - MII indicator register

This register indicates the statuses of SCAN command execution and MII management interface access.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + $00A4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	NVALID	SCANA	BUSY
R	R	R	R	R	R	R	R

Table 23-28 ETHAnMIND register contents

Bit position	Bit name	Function
2	NVALID	SCAN command start status 1: The SCAN command is under execution and the first read access has not finished. 0: Normal status
1	SCANA	SCAN command active 1: The SCAN command is under execution. 0: Normal status
0	BUSY	BUSY 1: The MII management interface is accessing an external PHY device. 0: The MII management interface is not accessing an external PHY device.

(17) ETHAnAFR - Address filter register

This register is used to set the conditions under which a receive packet is received.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $00C8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 4 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	PRO	PRM	AMC	ABC
R	R	R	R	R/W	R/W	R/W	R/W

Table 23-29 ETHAnAFR register contents

Bit position	Bit name	Function
3	PRO	Promiscuous mode All packets are valid in this mode.
2	PRM	Multicast reception In this mode, all multicast packets are valid and other packets are discarded.
1	AMC	Conditional multicast reception In this mode, multicast packets that satisfy the conditions are valid and other packets are discarded. Only multicast packets whose multicast address matches the values of the hash table specified by the ETHAnHT1 and ETHAnHT2 registers are received. The hash table is specified by the ETHAnHT1 and ETHAnHT2 registers.
0	ABC	Broadcast reception In this mode, broadcast packets are valid and other packets are discarded.

For details of the settings of the ETHAnAFR register and the packets to be filtered, see *Table 23-117 “Settings of ETHAnAFR register and packets to be filtered” on page 1759*.

(18) ETHAnHT1 - Hash table register 1

This register is used to specify the hash table used for conditional multicast packet detection.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $00CC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
HT131	HT130	HT129	HT128	HT127	HT126	HT125	HT124
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
HT123	HT122	HT121	HT120	HT119	HT118	HT11	HT116
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
HT115	HT114	HT113	HT112	HT111	HT110	HT19	HT18
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
HT17	HT16	HT15	HT14	HT13	HT12	HT11	HT10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-30 ETHAnHT1 register contents

Bit position	Bit name	Function
31 to 0	HT1[31:0]	Hash table 1 The hash table is used for conditional multicast packet detection. These bits indicate the higher 32 bits of the hash table. HT (63:32)

(19) ETHAnHT2 - Hash table register 2

This register is used to specify the hash table used for conditional multicast packet detection.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $00D0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
HT231	HT230	HT229	HT228	HT227	HT226	HT225	HT224
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
HT223	HT222	HT221	HT220	HT219	HT218	HT217	HT216
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
HT215	HT214	HT213	HT212	HT211	HT210	HT29	HT28
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
HT27	HT26	HT25	HT24	HT23	HT22	HT21	HT20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-31 ETHAnHT2 register contents

Bit position	Bit name	Function
31 to 0	HT2[31:0]	Hash table 2 The hash table is used for conditional multicast packet detection. These bits indicate the lower 32 bits of the hash table. HT (31:0)

(20) ETHAnCAR1 - Carry register 1

This register indicates that a statistics counter has overflowed. Each bit of this register corresponds to a statistics counter. When a statistics counter overflows, the corresponding bit in this register is set.

Each bit is cleared when it is read.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $00DC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
C1VT	C1UT	C1BT	C1MT	C1PT	C1TB	C1MX	C11K
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
C1FE	C1TF	C1OT	C1SF	C1BR	C1MR	C1PR	C1RB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-32 ETHAnCAR1 register contents

Bit position	Bit name	Function
15	C1VT	Overflow of ETHAnRVBT counter 0: Counter did not overflow. 1: Counter overflowed.
14	C1UT	Overflow of ETHAnTUCA counter 0: Counter did not overflow. 1: Counter overflowed.
13	C1BT	Overflow of ETHAnTBCA counter 0: Counter did not overflow. 1: Counter overflowed.
12	C1MT	Overflow of ETHAnTMCA counter 0: Counter did not overflow. 1: Counter overflowed.
11	C1PT	Overflow of ETHAnTPKT counter 0: Counter did not overflow. 1: Counter overflowed.
10	C1TB	Overflow of ETHAnTBYT counter 0: Counter did not overflow. 1: Counter overflowed.
9	C1MX	Overflow of ETHAnRMAX counter 0: Counter did not overflow. 1: Counter overflowed.
8	C11K	Overflow of ETHAnR1K counter 0: Counter did not overflow. 1: Counter overflowed.
7	C1FE	Overflow of ETHAnR511 counter 0: Counter did not overflow. 1: Counter overflowed.
6	C1TF	Overflow of ETHAnR255 counter 0: Counter did not overflow. 1: Counter overflowed.
5	C1OT	Overflow of ETHAnR127 counter 0: Counter did not overflow. 1: Counter overflowed.
4	C1SF	Overflow of ETHAnR64 counter 0: Counter did not overflow. 1: Counter overflowed.
3	C1BR	Overflow of ETHAnRBCA counter 0: Counter did not overflow. 1: Counter overflowed.
2	C1MR	Overflow of ETHAnRMCA counter 0: Counter did not overflow. 1: Counter overflowed.
1	C1PR	Overflow of ETHAnRPKT counter 0: Counter did not overflow. 1: Counter overflowed.
0	C1RB	Overflow of ETHAnRBYT counter 0: Counter did not overflow. 1: Counter overflowed.

(21) ETHAnCAR2 - Carry register 2

This register indicates that a statistics counter has overflowed. Each bit of this register corresponds to a statistics counter. When a statistics counter overflows, the corresponding bit in this register is set.

Each bit is cleared when it is read.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $00E0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 30 to 23 to “0”.

31	30	29	28	27	26	25	24
C2DV	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	C2IM	C2CS	C2NC	C2XC	C2LC	C2MC	C2SC
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
C2XD	C2DF	C2XF	C2TE	C2JB	C2FG	C2OV	C2UN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
C2FC	C2CD	C2FO	C2AL	C2UO	C2PF	C2CF	C2RE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-33 ETHAnCAR2 register contents (1/2)

Bit position	Bit name	Function
31	C2DV	Overrunning of status vector 0: Status vector does not overrun. 1: Status vector overruns.
22	C2IM	Overflow of ETHAnTIME counter 0: Counter did not overflow. 1: Counter overflowed.
21	C2CS	Overflow of ETHAnTCSE counter 0: Counter did not overflow. 1: Counter overflowed.
20	C2NC	Overflow of ETHAnTNCL counter 0: Counter did not overflow. 1: Counter overflowed.
19	C2XC	Overflow of ETHAnTXCL counter 0: Counter did not overflow. 1: Counter overflowed.
18	C2LC	Overflow of ETHAnTLCL counter 0: Counter did not overflow. 1: Counter overflowed.
17	C2MC	Overflow of ETHAnTMCL counter 0: Counter did not overflow. 1: Counter overflowed.
16	C2SC	Overflow of ETHAnTSCL counter 0: Counter did not overflow. 1: Counter overflowed.
15	C2XD	Overflow of ETHAnTXDF counter 0: Counter did not overflow. 1: Counter overflowed.
14	C2DF	Overflow of ETHAnTDFR counter 0: Counter did not overflow. 1: Counter overflowed.
13	C2XF	Overflow of ETHAnTXPF counter 0: Counter did not overflow. 1: Counter overflowed.
12	C2TE	Overflow of ETHAnTFCS counter 0: Counter did not overflow. 1: Counter overflowed.
11	C2JB	Overflow of ETHAnRJBR counter 0: Counter did not overflow. 1: Counter overflowed.
10	C2FG	Overflow of ETHAnRFRG counter 0: Counter did not overflow. 1: Counter overflowed.
9	C2OV	Overflow of ETHAnROVR counter 0: Counter did not overflow. 1: Counter overflowed.
8	C2UN	Overflow of ETHAnRUND counter 0: Counter did not overflow. 1: Counter overflowed.
7	C2FC	Overflow of ETHAnRFCR counter 0: Counter did not overflow. 1: Counter overflowed.

Table 23-33 ETHAnCAR2 register contents (2/2)

Bit position	Bit name	Function
6	C2CD	Overflow of ETHAnRCDE counter 0: Counter did not overflow. 1: Counter overflowed.
5	C2FO	Overflow of ETHAnRFLR counter 0: Counter did not overflow. 1: Counter overflowed.
4	C2AL	Overflow of ETHAnRALN counter 0: Counter did not overflow. 1: Counter overflowed.
3	C2UO	Overflow of ETHAnRXUO counter 0: Counter did not overflow. 1: Counter overflowed.
2	C2PF	Overflow of ETHAnRXPF counter 0: Counter did not overflow. 1: Counter overflowed.
1	C2CF	Overflow of ETHAnRXCF counter 0: Counter did not overflow. 1: Counter overflowed.
0	C2RE	Overflow of ETHAnRFCS counter 0: Counter did not overflow. 1: Counter overflowed.

(22) ETHAnCAM1 - Carry mask register 1

This register is used to mask the INTETMOV signal that is generated when a statistics counter has overflowed and the corresponding bit in the ETHAnCAR1 register has been set.

Each bit of this register can be masked.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0130_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
M1VT	M1UT	M1BT	M1MT	M1PT	M1TB	M1MX	M11K
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
M1FE	M1TF	M1OT	M1SF	M1BR	M1MR	M1PR	M1RB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-34 ETHAnCAM1 register contents

Bit position	Bit name	Function
15	M1VT	ETHAnRVBT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
14	M1UT	ETHAnTUCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
13	M1BT	ETHAnTBCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
12	M1MT	ETHAnTMCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
11	M1PT	ETHAnTPKT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
10	M1TB	ETHAnTBYT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
9	M1MX	ETHAnRMAX counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
8	M11K	ETHAnR1K counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
7	M1FE	ETHAnR511 counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
6	M1TF	ETHAnR255 counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
5	M1OT	ETHAnR127 counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
4	M1SF	ETHAnR64 counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
3	M1BR	ETHAnRBCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
2	M1MR	ETHAnRMCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
1	M1PR	ETHAnRPKT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
0	M1RB	ETHAnRBYT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).

(23) ETHAnCAM2 - Carry mask register 2

This register is used to mask the CAINT signal that is generated when a statistics counter has overflowed and the corresponding bit in the ETHAnCAR2 register has been set.

Each bit of this register can be masked.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0134_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 30 to 23 to “0”.

31	30	29	28	27	26	25	24
M2DV	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	M2IM	M2CS	M2NC	M2XC	M2LC	M2MC	M2SC
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
M2XD	M2DF	M2XF	M2TE	M2JB	M2FG	M2OV	M2UN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
M2FC	M2CD	M2FO	M2AL	M2UO	M2PF	M2CF	M2RE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-35 ETHAnCAM2 register contents (1/2)

Bit position	Bit name	Function
31	M2DV	Status vector overrun interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
22	M2IM	ETHAnTIME counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
21	M2CS	ETHAnTCSE counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
20	M2NC	ETHAnTNCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
19	M2XC	ETHAnTXCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
18	M2LC	ETHAnTLCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
17	M2MC	ETHAnTMCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
16	M2SC	ETHAnTSCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
15	M2XD	ETHAnTXDF counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
14	M2DF	ETHAnTDFR counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
13	M2XF	ETHAnTXPF counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
12	M2TE	ETHAnTFCS counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
11	M2JB	ETHAnRJBR counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
10	M2FG	ETHAnRFRG counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
9	M2OV	ETHAnROVR counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
8	M2UN	ETHAnRUND counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
7	M2FC	ETHAnRFCR counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).

Table 23-35 ETHAnCAM2 register contents (2/2)

Bit position	Bit name	Function
6	M2CD	ETHAnRCDE counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
5	M2FO	ETHAnRFLR counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
4	M2AL	ETHAnRALN counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
3	M2UO	ETHAnRXUO counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
2	M2PF	ETHAnRXPF counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
1	M2CF	ETHAnRXCF counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
0	M2RE	ETHAnRFCS counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).

23.4.2 Statistics counters

(1) ETHAnRBYT - Reception byte counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0140_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RBYT31	RBYT30	RBYT29	RBYT28	RBYT27	RBYT26	RBYT25	RBYT24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RBYT23	RBYT22	RBYT21	RBYT20	RBYT19	RBYT18	RBYT17	RBYT16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RBYT15	RBYT14	RBYT13	RBYT12	RBYT11	RBYT10	RBYT9	RBYT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RBYT7	RBYT6	RBYT5	RBYT4	RBYT3	RBYT2	RBYT1	RBYT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-36 ETHAnRBYT register contents

Bit position	Bit name	Function
31 to 0	RBYT[31:0]	This counter indicates the number of bytes in a received packet. It counts from the destination address byte to the FCS byte. It continues counting bytes even if an error occurs. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.HUGEN bit is 0, this counter is incremented because the packet length is assumed to be that specified by the ETHAnLMAX register.

(2) ETHAnRPKT - Reception packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0144_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RPKT31	RPKT30	RPKT29	RPKT28	RPKT27	RPKT26	RPKT25	RPKT24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RPKT23	RPKT22	RPKT21	RPKT20	RPKT19	RPKT18	RPKT1	RPKT16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RPKT15	RPKT14	RPKT13	RPKT12	RPKT11	RPKT10	RPKT9	RPKT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RPKT7	RPKT6	RPKT5	RPKT4	RPKT3	RPKT2	RPKT1	RPKT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-37 ETHAnRPKT register contents

Bit position	Bit name	Function
31 to 0	RPKT[31:0]	This counter is incremented when any packet, including packets in which an error has occurred, all unicast packets, all multicast packets, and broadcast packets, is received.

(3) ETHAnRFCS - Reception FCS error frame counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0148_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RFCS31	RFCS30	RFCS29	RFCS28	RFCS27	RFCS26	RFCS25	RFCS24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RFCS23	RFCS22	RFCS21	RFCS20	RFCS19	RFCS18	RFCS17	RFCS16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RFCS15	RFCS14	RFCS13	RFCS12	RFCS11	RFCS10	RFCS9	RFCS8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RFCS7	RFCS6	RFCS5	RFCS4	RFCS3	RFCS2	RFCS1	RFCS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-38 ETHAnRFCS register contents

Bit position	Bit name	Function
31 to 0	RFCS[31:0]	This counter is incremented if a CRC error occurs in a receive packet. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.HUGEN bit is 0, a CRC check is executed when the packet length reaches the value specified by the ETHAnLMAX register, so this counter might be incremented because the received packet is assumed to contain a CRC error.

(4) ETHAnRMCA - Reception multicast packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $014C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RMCA31	RMCA30	RMCA29	RMCA28	RMCA27	RMCA26	RMCA25	RMCA24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RMCA23	RMCA22	RMCA21	RMCA20	RMCA19	RMCA18	RMCA17	RMCA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RMCA15	RMCA14	RMCA13	RMCA12	RMCA11	RMCA10	RMCA9	RMCA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RMCA7	RMCA6	RMCA5	RMCA4	RMCA3	RMCA2	RMCA1	RMCA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-39 ETHAnRMCA register contents

Bit position	Bit name	Function
31 to 0	RMCA[31:0]	This counter is incremented when a multicast packet whose length is 64 to 1,518 bytes (64 to 1,522 bytes for a VLAN frame) is received. It is not incremented when a multicast packet in which a CRC error has occurred or a broadcast packet is received.

(5) ETHAnRBCA - Reception broadcast packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0150_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RBCA31	RBCA30	RBCA29	RBCA28	RBCA27	RBCA26	RBCA25	RBCA24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RBCA23	RBCA22	RBCA21	RBCA20	RBCA19	RBCA18	RBCA17	RBCA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RBCA15	RBCA14	RBCA13	RBCA12	RBCA11	RBCA10	RBCA9	RBCA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RBCA7	RBCA6	RBCA5	RBCA4	RBCA3	RBCA2	RBCA1	RBCA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-40 ETHAnRBCA register contents

Bit position	Bit name	Function
31 to 0	RBCA[31:0]	This counter is incremented when a broadcast packet whose length is 64 to 1,518 bytes (64 to 1,522 bytes for a VLAN frame) is received. It is not incremented when a broadcast packet in which a CRC error has occurred or a multicast packet is received.

(6) ETHAnRXCF - Reception control frame packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0154_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RXCF31	RXCF30	RXCF29	RXCF28	RXCF27	RXCF26	RXCF25	RXCF24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RXCF23	RXCF22	RXCF21	RXCF20	RXCF19	RXCF18	RXCF17	RXCF16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RXCF15	RXCF14	RXCF13	RXCF12	RXCF11	RXCF10	RXCF9	RXCF8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RXCF7	RXCF6	RXCF5	RXCF4	RXCF3	RXCF2	RXCF1	RXCF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-41 ETHAnRXCF register contents

Bit position	Bit name	Function
31 to 0	RXCF[31:0]	This counter is incremented when any control frame, including pause control frames and unsupported control frames is received. It is not incremented when a control frame in which a CRC error has been detected is received.

(7) ETHAnRXPF - Reception pause frame packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0158_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RXPF31	RXPF30	RXPF29	RXPF28	RXPF27	RXPF26	RXPF25	RXPF24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RXPF23	RXPF22	RXPF21	RXPF20	RXPF19	RXPF18	RXPF17	RXPF16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RXPF15	RXPF14	RXPF13	RXPF12	RXPF11	RXPF10	RXPF9	RXPF8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RXPF7	RXPF6	RXPF5	RXPF4	RXPF3	RXPF2	RXPF1	RXPF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-42 ETHAnRXPF register contents

Bit position	Bit name	Function
31 to 0	RXPF[31:0]	This counter is incremented when a valid pause control frame is received.

(8) ETHAnRXUO - Reception undefined control packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $015C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RXUO31	RXUO30	RXUO29	RXUO28	RXUO27	RXUO26	RXUO25	RXUO24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RXUO23	RXUO22	RXUO21	RXUO20	RXUO19	RXUO18	RXUO17	RXUO16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RXUO15	RXUO14	RXUO13	RXUO12	RXUO11	RXUO10	RXUO9	RXUO8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RXUO7	RXUO6	RXUO5	RXUO4	RXUO3	RXUO2	RXUO1	RXUO0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-43 ETHAnRXUO register contents

Bit position	Bit name	Function
31 to 0	RXUO[31:0]	This counter is incremented when a control frame that has an opcode other than PAUSE or a pause control frame that has an invalid destination address is received. It is not incremented when a pause control frame in which a CRC error has been detected is received.

(9) ETHAnRALN - Reception alignment error counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0160_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RALN31	RALN30	RALN29	RALN28	RALN27	RALN26	RALN25	RALN24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RALN23	RALN22	RALN21	RALN20	RALN19	RALN18	RALN17	RALN16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RALN15	RALN14	RALN13	RALN12	RALN11	RALN10	RALN9	RALN8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RALN7	RALN6	RALN5	RALN4	RALN3	RALN2	RALN1	RALN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-44 ETHAnRALN register contents

Bit position	Bit name	Function
31 to 0	RALN[31:0]	This counter is incremented if a CRC error and a dribble nibble occur in a received packet. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.HUGEN bit is 0, an alignment error check is executed when the packet length reaches the length (in bytes) specified by the ETHAnLMAX register, so this counter is not incremented.

(10) ETHAnRFLR - Reception frame length error counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0164_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RFLR31	RFLR30	RFLR29	RFLR28	RFLR27	RFLR26	RFLR25	RFLR24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RFLR23	RFLR22	RFLR21	RFLR20	RFLR19	RFLR18	RFLR17	RFLR16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RFLR15	RFLR14	RFLR13	RFLR12	RFLR11	RFLR10	RFLR9	RFLR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RFLR7	RFLR6	RFLR5	RFLR4	RFLR3	RFLR2	RFLR1	RFLR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-45 ETHAnRFLR register contents

Bit position	Bit name	Function
31 to 0	RFLR[31:0]	This counter is incremented if the value of the length field of a receive packet does not match the data field length of a packet actually received. If the value of the length field is 1,501 or more (for example, when the bytes equivalent to the length field are used as the Ethernet type field), this counter is not incremented.

(11) ETHAnRCDE - Reception code error counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0168_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RCDE31	RCDE30	RCDE29	RCDE28	RCDE27	RCDE26	RCDE25	RCDE24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RCDE23	RCDE22	RCDE21	RCDE20	RCDE19	RCDE18	RCDE17	RCDE16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RCDE15	RCDE14	RCDE13	RCDE12	RCDE11	RCDE10	RCDE9	RCDE8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RCDE7	RCDE6	RCDE5	RCDE4	RCDE3	RCDE2	RCDE1	RCDE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-46 ETHAnRCDE register contents

Bit position	Bit name	Function
31 to 0	RCDE[31:0]	This counter is incremented if an illegal data symbol has been detected at least once while a carrier is being detected.

(12) ETHAnRFCR - Reception false carrier counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $016C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RFCR31	RFCR30	RFCR29	RFCR28	RFCR27	RFCR26	RFCR25	RFCR24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RFCR23	RFCR22	RFCR21	RFCR20	RFCR19	RFCR18	RFCR17	RFCR16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RFCR15	RFCR14	RFCR13	RFCR12	RFCR11	RFCR10	RFCR9	RFCR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RFCR7	RFCR6	RFCR5	RFCR4	RFCR3	RFCR2	RFCR1	RFCR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-47 ETHAnRFCR register contents

Bit position	Bit name	Function
31 to 0	RFCR[31:0]	If a false carrier is generated in the idle status, this counter is incremented after the next packet is received. It is assumed that a false carrier has occurred if 1110_B is input as nibble data from RXD while RXER is high. This counter is incremented only once even if multiple false carriers are generated in the idle status.

(13) ETHAnRUND - Reception undersize packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0170_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RUND31	RUND30	RUND29	RUND28	RUND27	RUND26	RUND25	RUND24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RUND23	RUND22	RUND21	RUND20	RUND19	RUND18	RUND1	RUND16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RUND15	RUND14	RUND13	RUND12	RUND11	RUND10	RUND9	RUND8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RUND7	RUND6	RUND5	RUND4	RUND3	RUND2	RUND1	RUND0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-48 ETHAnRUND register contents

Bit position	Bit name	Function
31 to 0	RUND[31:0]	This counter is incremented if the receive packet length is less than 64 bytes and the packet contains a valid FCS field.

(14) ETHAnROVR - Reception oversize packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0174_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ROVR31	ROVR30	ROVR29	ROVR28	ROVR27	ROVR26	ROVR25	ROVR24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ROVR23	ROVR22	ROVR21	ROVR20	ROVR19	ROVR18	ROVR17	ROVR16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ROVR15	ROVR14	ROVR13	ROVR12	ROVR11	ROVR10	ROVR9	ROVR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ROVR7	ROVR6	ROVR5	ROVR4	ROVR3	ROVR2	ROVR1	ROVR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-49 ETHAnROVR register contents

Bit position	Bit name	Function
31 to 0	ROVR[31:0]	This counter is incremented if the receive packet length exceeds 1,518 bytes (1,522 bytes for a VLAN frame) and the packet contains a valid FCS field. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.HUGEN bit is 0, a CRC check is executed when the packet length reaches the value specified by the ETHAnLMAX register. Therefore, a CRC error might be detected and this counter might not be incremented.

(15) ETHAnRFRG - Reception fragment counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0178_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RFRG31	RFRG30	RFRG29	RFRG28	RFRG27	RFRG26	RFRG25	RFRG24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RFRG23	RFRG22	RFRG21	RFRG20	RFRG19	RFRG18	RFRG17	RFRG16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RFRG15	RFRG14	RFRG13	RFRG12	RFRG11	RFRG10	RFRG9	RFRG8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RFRG7	RFRG6	RFRG5	RFRG4	RFRG3	RFRG2	RFRG1	RFRG0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-50 ETHAnRFRG register contents

Bit position	Bit name	Function
31 to 0	RFRG[31:0]	This counter is incremented if the receive packet length is less than 64 bytes and the packet contains a CRC error or an alignment error.

(16) ETHAnRJBR - Reception jabber counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $017C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RJBR31	RJBR30	RJBR29	RJBR28	RJBR27	RJBR26	RJBR25	RJBR24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RJBR23	RJBR22	RJBR21	RJBR20	RJBR19	RJBR18	RJBR17	RJBR16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RJBR15	RJBR14	RJBR13	RJBR12	RJBR11	RJBR10	RJBR9	RJBR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RJBR7	RJBR6	RJBR5	RJBR4	RJBR3	RJBR2	RJBR1	RJBR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-51 ETHAnRJBR register contents

Bit position	Bit name	Function
31 to 0	RJBR[31:0]	This counter is incremented if the receive packet length exceeds 1,518 bytes (1,522 bytes for a VLAN frame) and the packet contains a CRC error or an alignment error. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.HUGEN bit is 0, a CRC check is executed when the packet length reaches the value specified by the ETHAnLMAX register. Therefore, a CRC error might be detected and this counter might be incremented.

(17) ETHAnR64 - Receive 64-byte frame counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0180_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
R6431	R6430	R6429	R6428	R6427	R6426	R6425	R6424
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
R6423	R6422	R6421	R6420	R6419	R6418	R6417	R6416
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
R6415	R6414	R6413	R6412	R6411	R6410	R649	R648
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
R647	R646	R645	R644	R643	R642	R641	R640
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-52 ETHAnR64 register contents

Bit position	Bit name	Function
31 to 0	R64[31:0]	This counter is incremented if the receive packet length is 64 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(18) ETHAnR127 - Receive 65- to 127-byte frame counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0184_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
R12731	R12730	R12729	R12728	R12727	R12726	R12725	R12724
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
R12723	R12722	R12721	R12720	R12719	R12718	R12717	R12716
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
R12715	R12714	R12713	R12712	R12711	R12710	R1279	R1278
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
R1277	R1276	R1275	R1274	R1273	R1272	R1271	R1270
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-53 ETHAnR127 register contents

Bit position	Bit name	Function
31 to 0	R127[31:0]	This counter is incremented if the receive packet length is 64 to 127 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(19) ETHAnR255 - Receive 128- to 255-byte frame counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0188_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
R25531	R25530	R25529	R25528	R25527	R25526	R25525	R25524
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
R25523	R25522	R25521	R25520	R25519	R25518	R25517	R25516
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
R25515	R25514	R25513	R25512	R25511	R25510	R2559	R2558
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
R2557	R2556	R2555	R2554	R2553	R2552	R2551	R2550
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-54 ETHAnR255 register contents

Bit position	Bit name	Function
31 to 0	R255[31:0]	This counter is incremented if the receive packet length is 128 to 255 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(20) ETHAnR511 - Receive 256- to 511-byte frame counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $018C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
R51131	R51130	R51129	R51128	R51127	R51126	R51125	R51124
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
R51123	R51122	R51121	R51120	R51119	R51118	R51117	R51116
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
R51115	R51114	R51113	R51112	R51111	R51110	R5119	R5118
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
R5117	R5116	R5115	R5114	R5113	R5112	R5111	R5110
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-55 ETHAnR511 register contents

Bit position	Bit name	Function
31 to 0	R511[31:0]	This counter is incremented if the receive packet length is 256 to 511 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(21) ETHAnR1K - Receive 512- to 1023-byte frame counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0190_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
R1K31	R1K30	R1K29	R1K28	R1K27	R1K26	R1K25	R1K24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
R1K23	R1K22	R1K21	R1K20	R1K19	R1K18	R1K17	R1K16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
R1K15	R1K14	R1K13	R1K12	R1K11	R1K10	R1K9	R1K8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
R1K7	R1K6	R1K5	R1K4	R1K3	R1K2	R1K1	R1K0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-56 ETHAnR1K register contents

Bit position	Bit name	Function
31 to 0	R1K[31:0]	This counter is incremented if the receive packet length is 512 to 1,023 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(22) ETHAnRMAX - Receive 1024- to RMAX-byte frame counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0194_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RMAX31	RMAX30	RMAX29	RMAX28	RMAX27	RMAX26	RMAX25	RMAX24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RMAX23	RMAX22	RMAX21	RMAX20	RMAX19	RMAX18	RMAX17	RMAX16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RMAX15	RMAX14	RMAX13	RMAX12	RMAX11	RMAX10	RMAX9	RMAX8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RMAX7	RMAX6	RMAX5	RMAX4	RMAX3	RMAX2	RMAX1	RMAX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-57 ETHAnRMAX register contents

Bit position	Bit name	Function
31 to 0	RMAX[31:0]	This counter is incremented if the receive packet length is 1,024 to 1,518 bytes (1,024 to 1,522 bytes for a VLAN frame). It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(23) ETHAnRVBT - Receive valid byte counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0198_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RVBT31	RVBT30	RVBT29	RVBT28	RVBT27	RVBT26	RVBT25	RVBT24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RVBT23	RVBT22	RVBT21	RVBT20	RVBT19	RVBT18	RVBT17	RVBT16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RVBT15	RVBT14	RVBT13	RVBT12	RVBT11	RVBT10	RVBT9	RVBT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RVBT7	RVBT6	RVBT5	RVBT4	RVBT3	RVBT2	RVBT1	RVBT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-58 ETHAnRVBT register contents

Bit position	Bit name	Function
31 to 0	RVBT[31:0]	This counter indicates the byte count of a valid packet. It counts from the destination address byte to the FCS byte.

(24) ETHAnTBYT - Transmission byte counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01C0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TBYT31	TBYT30	TBYT29	TBYT28	TBYT27	TBYT26	TBYT25	TBYT24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TBYT23	TBYT22	TBYT21	TBYT20	TBYT19	TBYT18	TBYT17	TBYT16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TBYT15	TBYT14	TBYT13	TBYT12	TBYT11	TBYT10	TBYT9	TBYT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TBYT7	TBYT6	TBYT5	TBYT4	TBYT3	TBYT2	TBYT1	TBYT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-59 ETHAnTBYT register contents

Bit position	Bit name	Function
31 to 0	TBYT[31:0]	This counter indicates the number of bytes in a transmit packet. When a collision occurs before transmission is finished or aborted, the transmission byte at which the collision occurred is also counted. The preamble and SFD are not included in the byte count indication.

(25) ETHAnTPKT - Transmission packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01C4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TPKT31	TPKT30	TPKT29	TPKT28	TPKT27	TPKT26	TPKT25	TPKT24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TPKT23	TPKT22	TPKT21	TPKT20	TPKT19	TPKT18	TPKT17	TPKT16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TPKT15	TPKT14	TPKT13	TPKT12	TPKT11	TPKT10	TPKT9	TPKT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TPKT7	TPKT6	TPKT5	TPKT4	TPKT3	TPKT2	TPKT1	TPKT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-60 ETHAnTPKT register contents

Bit position	Bit name	Function
31 to 0	TPKT[31:0]	This counter is incremented when any packet is transmitted. This includes when a packet in which an error has occurred, unicast packet, multicast packet, or broadcast packet is transmitted.

(26) ETHAnTFCS - Transmission FCS error frame counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01C8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TFCS31	TFCS30	TFCS29	TFCS28	TFCS27	TFCS26	TFCS25	TFCS24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TFCS23	TFCS22	TFCS21	TFCS20	TFCS19	TFCS18	TFCS17	TFCS16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TFCS15	TFCS14	TFCS13	TFCS12	TFCS11	TFCS10	TFCS9	TFCS8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TFCS7	TFCS6	TFCS5	TFCS4	TFCS3	TFCS2	TFCS1	TFCS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-61 ETHAnTFCS register contents

Bit position	Bit name	Function
31 to 0	TFCS[31:0]	This counter is incremented if a CRC error is detected in the FCS field that is appended to a transmit packet. It is not incremented if transmission is aborted.

(27) ETHAnTMCA - Transmission multicast packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01CC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TMCA31	TMCA30	TMCA29	TMCA28	TMCA27	TMCA26	TMCA25	TMCA24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TMCA23	TMCA22	TMCA21	TMCA20	TMCA19	TMCA18	TMCA17	TMCA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TMCA15	TMCA14	TMCA13	TMCA12	TMCA11	TMCA10	TMCA9	TMCA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TMCA7	TMCA6	TMCA5	TMCA4	TMCA3	TMCA2	TMCA1	TMCA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-62 ETHAnTMCA register contents

Bit position	Bit name	Function
31 to 0	TMCA[31:0]	This counter is incremented when a multicast packet is transmitted. It is not incremented when a broadcast packet is transmitted. Nor is it incremented if transmission is aborted or if a CRC error is detected.

(28) ETHAnTBCA - Transmission broadcast packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01D0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TBCA31	TBCA30	TBCA29	TBCA28	TBCA27	TBCA26	TBCA25	TBCA24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TBCA23	TBCA22	TBCA21	TBCA20	TBCA19	TBCA18	TBCA17	TBCA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TBCA15	TBCA14	TBCA13	TBCA12	TBCA11	TBCA10	TBCA9	TBCA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TBCA7	TBCA6	TBCA5	TBCA4	TBCA3	TBCA2	TBCA1	TBCA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-63 ETHAnTBCA register contents

Bit position	Bit name	Function
31 to 0	TBCA[31:0]	This counter is incremented when a broadcast packet is transmitted. It is not incremented when a multicast packet is transmitted. Nor is it incremented if transmission is aborted or if a CRC error is detected.

(29) ETHAnTUCA - Transmit unicast packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01D4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TUCA31	TUCA30	TUCA29	TUCA28	TUCA27	TUCA26	TUCA25	TUCA24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TUCA23	TUCA22	TUCA21	TUCA20	TUCA19	TUCA18	TUCA17	TUCA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TUCA15	TUCA14	TUCA13	TUCA12	TUCA11	TUCA10	TUCA9	TUCA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TUCA7	TUCA6	TUCA5	TUCA4	TUCA3	TUCA2	TUCA1	TUCA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-64 ETHAnTUCA register contents

Bit position	Bit name	Function
31 to 0	TUCA[31:0]	This counter is incremented when a unicast packet is transmitted. It is not incremented if transmission is aborted or if a CRC error is detected.

(30) ETHAnTXPF - Transmission pause control frame counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01D8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TXPF31	TXPF30	TXPF29	TXPF28	TXPF27	TXPF26	TXPF25	TXPF24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TXPF23	TXPF22	TXPF21	TXPF20	TXPF19	TXPF18	TXPF17	TXPF16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TXPF15	TXPF14	TXPF13	TXPF12	TXPF11	TXPF10	TXPF9	TXPF8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TXPF7	TXPF6	TXPF5	TXPF4	TXPF3	TXPF2	TXPF1	TXPF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-65 ETHAnTXPF register contents

Bit position	Bit name	Function
31 to 0	TXPF[31:0]	This counter is incremented each time a pause control frame is transmitted when the maximum amount of data has been stored in the receive FIFO.

(31) ETHAnTDFR - Transmission delay packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01DC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TDFR31	TDFR30	TDFR29	TDFR28	TDFR27	TDFR26	TDFR25	TDFR24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TDFR23	TDFR22	TDFR21	TDFR20	TDFR19	TDFR18	TDFR17	TDFR16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TDFR15	TDFR14	TDFR13	TDFR12	TDFR11	TDFR10	TDFR9	TDFR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TDFR7	TDFR6	TDFR5	TDFR4	TDFR3	TDFR2	TDFR1	TDFR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-66 ETHAnTDFR register contents

Bit position	Bit name	Function
31 to 0	TDFR[31:0]	This counter is incremented if a transmit delay occurs because of carrier detection when transmission is about to start. It is not incremented if a collision occurs during transmission that was started after the delay occurred.

(32) ETHAnTXDF - Transmission excessive delay packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01E0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TXDF31	TXDF30	TXDF29	TXDF28	TXDF27	TXDF26	TXDF25	TXDF24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TXDF23	TXDF22	TXDF21	TXDF20	TXDF19	TXDF18	TXDF17	TXDF16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TXDF15	TXDF14	TXDF13	TXDF12	TXDF11	TXDF10	TXDF9	TXDF8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TXDF7	TXDF6	TXDF5	TXDF4	TXDF3	TXDF2	TXDF1	TXDF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-67 ETHAnTXDF register contents

Bit position	Bit name	Function
31 to 0	TXDF[31:0]	This counter is incremented if transmission is aborted by an excessive delay.

(33) ETHAnTSCL - Transmission single collision packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01E4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TSCL31	TSCL30	TSCL29	TSCL28	TSCL27	TSCL26	TSCL25	TSCL24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TSCL23	TSCL22	TSCL21	TSCL20	TSCL19	TSCL18	TSCL17	TSCL16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TSCL15	TSCL14	TSCL13	TSCL12	TSCL11	TSCL10	TSCL9	TSCL8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TSCL7	TSCL6	TSCL5	TSCL4	TSCL3	TSCL2	TSCL1	TSCL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-68 ETHAnTSCL register contents

Bit position	Bit name	Function
31 to 0	TSCL[31:0]	This counter is incremented if a transmission finishes successfully after a single collision occurred during the transmission.

(34) ETHAnTMCL - Transmission multiple collision packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01E8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TMCL31	TMCL30	TMCL29	TMCL28	TMCL27	TMCL26	TMCL25	TMCL24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TMCL23	TMCL22	TMCL21	TMCL20	TMCL19	TMCL18	TMCL17	TMCL16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TMCL15	TMCL14	TMCL13	TMCL12	TMCL11	TMCL10	TMCL9	TMCL8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TMCL7	TMCL6	TMCL5	TMCL4	TMCL3	TMCL2	TMCL1	TMCL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-69 ETHAnTMCL register contents

Bit position	Bit name	Function
31 to 0	TMCL[31:0]	This counter is incremented if a transmission finishes successfully after a collision occurred multiple times (but less times than the value specified by ETHAnCLRT.RETRY) during the transmission.

(35) ETHAnTLCL - Transmission late collision packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01EC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TLCL31	TLCL30	TLCL29	TLCL28	TLCL27	TLCL26	TLCL25	TLCL24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TLCL23	TLCL22	TLCL21	TLCL20	TLCL19	TLCL18	TLCL17	TLCL16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TLCL15	TLCL14	TLCL13	TLCL12	TLCL11	TLCL10	TLCL9	TLCL8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TLCL7	TLCL6	TLCL5	TLCL4	TLCL3	TLCL2	TLCL1	TLCL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-70 ETHAnTLCL register contents

Bit position	Bit name	Function
31 to 0	TLCL[31:0]	This counter is incremented if a late collision occurs during transmission.

(36) ETHAnTXCL - Transmission excessive collision packet counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01F0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TXCL31	TXCL30	TXCL29	TXCL28	TXCL27	TXCL26	TXCL25	TXCL24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TXCL23	TXCL22	TXCL21	TXCL20	TXCL19	TXCL18	TXCL17	TXCL16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TXCL15	TXCL14	TXCL13	TXCL12	TXCL11	TXCL10	TXCL9	TXCL8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TXCL7	TXCL6	TXCL5	TXCL4	TXCL3	TXCL2	TXCL1	TXCL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-71 ETHAnTXCL register contents

Bit position	Bit name	Function
31 to 0	TXCL[31:0]	This counter is incremented if collision occurs more than the times specified by ETHAnCLRT.RETRY in a single transmission.

(37) ETHAnTNCL - Transmission total collision counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01F4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TNCL31	TNCL30	TNCL29	TNCL28	TNCL27	TNCL26	TNCL25	TNCL24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TNCL23	TNCL22	TNCL21	TNCL20	TNCL19	TNCL18	TNCL17	TNCL16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TNCL15	TNCL14	TNCL13	TNCL12	TNCL11	TNCL10	TNCL9	TNCL8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TNCL7	TNCL6	TNCL5	TNCL4	TNCL3	TNCL2	TNCL1	TNCL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-72 ETHAnTNCL register contents

Bit position	Bit name	Function
31 to 0	TNCL[31:0]	This counter counts the number of collisions after which transmission finishes successfully.

(38) ETHAnTCSE - Transmission carrier sense error counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01F8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TCSE31	TCSE30	TCSE29	TCSE28	TCSE27	TCSE26	TCSE25	TCSE24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TCSE23	TCSE22	TCSE21	TCSE20	TCSE19	TCSE18	TCSE17	TCSE16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TCSE15	TCSE14	TCSE13	TCSE12	TCSE11	TCSE10	TCSE9	TCSE8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TCSE7	TCSE6	TCSE5	TCSE4	TCSE3	TCSE2	TCSE1	TCSE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-73 ETHAnTCSE register contents

Bit position	Bit name	Function
31 to 0	TCSE[31:0]	This counter is incremented if a carrier sense error occurs during transmission.

(39) ETHAnTIME - MAC internal error counter

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $01FC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TIME31	TIME30	TIME29	TIME28	TIME27	TIME26	TIME25	TIME24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TIME23	TIME22	TIME21	TIME20	TIME19	TIME18	TIME17	TIME16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TIME15	TIME14	TIME13	TIME12	TIME11	TIME10	TIME9	TIME8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TIME7	TIME6	TIME5	TIME4	TIME3	TIME2	TIME1	TIME0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-74 ETHAnTIME register contents

Bit position	Bit name	Function
31 to 0	TIME[31:0]	This counter is incremented if an error occurs in MAC during transmission or if a packet longer than the value specified by the ETHAnLMAX register is transmitted.

23.4.3 FIFO controller control registers

(1) ETHAnMFFCONT - FIFO controller control register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0200_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. Be sure to set the following bits to the values specified below (fixed values). If other values are set, the correct operation cannot be guaranteed.
   - RXSDMA[1:0] = 10
   - ASOE = 0
   - APS = 1
   - APL = 1
   - RXTHRC = 0
   - TXTHRC = 0
2. Be sure to set bits 29, 28, 23 to 19, 13, and 7 to 3 to “0”.

31	30	29	28	27	26	25	24
LOOPBACK	RCSEL	0	0	IMLP3	IMLP2	IMLP1	IMLP0
R/W	R/W	R	R	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	FLOWCNT	IVPAUSE	ZEROPAUSE
R	R	R	R	R	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RXSDMA1	RXSDMA0	0	ASOE	APS	APL	RXTHRC	RXEN
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	TABT	TXTHRC	TXEN
R	R	R	R	R	R/W	R/W	R/W

Table 23-75 ETHAnMFFCONT register contents

Bit position	Bit name	Function
31	LOOPBACK	Loopback mode Loopback between the transmit FIFO and receive FIFO is performed. 0: Normal mode 1: Loopback mode
30	RCSEL	RXCLK selection TXCLK is selected to be internally connected, instead of RXCLK. Specify a value for this bit if it is necessary to switch RXCLK to TXCLK when the MAC or the FIFO controller is in the loopback mode. 0: Normal mode 1: Clock switch mode (RXCLK switched to TXCLK)
27 to 24	IMLP[3:0]	Set these bits to 0000.
18	FLOWCNT	Flow control enable/disable 0: Flow control is disabled. 1: Flow control is enabled.
17	IVPAUSE	Interval pause packet transmission control This bit specifies the method of retransmitting a pause packet. 0: Retransmission by threshold value of FIFO Internal pause timer (ETHAnPAUSETM.IPTIME) is not used. 1: Retransmission by internal pause timer Internal pause timer (ETHAnPAUSETM.IPTIME) is used.
16	ZEROPAUSE	Zero pause packet output enable/disable 0: Zero pause packet output is disabled. 1: Zero pause packet output is enabled.
15, 14	RXSDMA[1:0]	Fix these bits to 10.
12	ASOE	Fix this bit to 0.
11	APS	Fix this bit to 1.
10	APL	Fix this bit to 1.
9	RXTHRC	Fix this bit to 0.
8	RXEN	Reception enable 0: Disables reception. 1: Enables reception. [Timing for disabling reception] If this bit is cleared to disable reception while a packet is being written from the MAC to the receive FIFO, the receive FIFO write circuit is stopped after the packet has been written. The receive FIFO of the system stops after all the packets written in the receive FIFO have been read. The flow control circuit is not stopped by this bit.
2	TABT	Transmission abort control This bit retransmits a packet that has been aborted by the MAC. 0: Discards the packet. 1: Retransmits the packet.
1	TXTHRC	Fix this bit to 0.
0	TXEN	Transmission enable 0: Disables transmission. 1: Enables transmission. [Timing for disabling transmission] If this bit is cleared to disable transmission while a packet is being written to the transmit FIFO, the transmit FIFO write circuit is stopped after the packet has been written (after the END flag has been set), cancelling a request to write the next packet. Packet transfer to the MAC is stopped after all the packets in the transmit FIFO have been transferred (after the empty status is indicated).

(2) ETHAnRSTCNT - Software reset control register

This register is used to control software reset.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0204_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 17, 15 to 9, and 7 to 1 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	RFFLSH
R	R	R	R	R	R	R	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	TFFLSH
R	R	R	R	R	R	R	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	SFTRST
R	R	R	R	R	R	R	R/W

Table 23-76 ETHAnRSTCNT register contents

Bit position	Bit name	Function
16	RFFLSH	Receive FIFO clear (flush) This bit clears the receive FIFO, reception control circuit, flow control circuit, reception status register, and interrupt registers related to reception. Writing 1 to this bit starts the reset operation, and then this bit is automatically cleared. This bit is always read as 0.
8	TFFLSH	Transmit FIFO clear (flush) This bit clears the transmit FIFO, transmission control circuit, transmission status register, and interrupt registers related to transmission. Writing 1 to this bit starts the reset operation, and then this bit is automatically cleared.
0	SFTRST	Software reset This bit resets all the circuits of the FIFO controller (MFF). Writing 1 to this bit starts the reset operation, and then this bit is automatically cleared.

(3) ETHA0FLOWTH - Flow control threshold value register

This register is used to specify a threshold value for the receive FIFO to start flow control, and a threshold value to transmit a zero pause control frame.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0218_H

Initial value $0600\ 0200_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 27 and 15 to 11 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	FLOWTHR10	FLOWTHR9	FLOWTHR8
R	R	R	R	R	R/W	R/W	R/W
23	22	21	20	19	18	17	16
FLOWTHR7	FLOWTHR6	FLOWTHR5	FLOWTHR4	FLOWTHR3	FLOWTHR2	FLOWTHR1	FLOWTHR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	ZPTHR10	ZPTHR9	ZPTHR8
R	R	R	R	R	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ZPTHR7	ZPTHR6	ZPTHR5	ZPTHR4	ZPTHR3	ZPTHR2	ZPTHR1	ZPTHR0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 23-77 ETHA0FLOWTH register contents

Bit position	Bit name	Function
26:16	FLOWTHR[10:0]	These bits specify the threshold value in bytes for the receive FIFO to start flow control. Flow control starts when the amount of data in the receive FIFO reaches the value specified by these bits. Back pressure transmission is executed in the half-duplex mode and pause control packets are transmitted in the full-duplex mode. Writing bit 17 or 16 is ignored because the receive FIFO can only be written in 32-bit (4-byte) units. Bits 17 and 16 are always read as 0.
10:0	ZPTHR[10:0]	These bits specify the threshold value in bytes for transmitting a zero pause control packet. When zero pause packet transmission is enabled by setting the MFFCNT.ZEROPAUSE bit to 1 (high level) and flow is controlled by pause control packets, a zero pause packet is transmitted if the amount of data in the receive FIFO falls below the threshold value specified by these bits. Writing bit 1 or 0 is ignored because the receive FIFO can only be written in 32-bit (4-byte) units. Bits 1 and 0 are always read as 0.

(4) ETHAnPAUSETM - Pause timer value register

This register is used to set the pause time.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $021C_H$

Initial value $7FFF\ FFFF_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
IPTIME15	IPTIME14	IPTIME13	IPTIME12	IPTIME11	IPTIME10	IPTIME9	IPTIME8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
IPTIME7	IPTIME6	IPTIME5	IPTIME4	IPTIME3	IPTIME2	IPTIME1	IPTIME0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
PAUSETM_MAX15	PAUSETM_MAX14	PAUSETM_MAX13	PAUSETM_MAX12	PAUSETM_MAX11	PAUSETM_MAX10	PAUSETM_MAX9	PAUSETM_MAX8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
PAUSETM_MAX7	PAUSETM_MAX6	PAUSETM_MAX5	PAUSETM_MAX4	PAUSETM_MAX3	PAUSETM_MAX2	PAUSETM_MAX1	PAUSETM_MAX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-78 ETHAnPAUSETM register contents

Bit position	Bit name	Function
31 to 16	IPTIME[15:0]	Interval pause packet timer value These bits specify the interval for transmitting a pause packet when interval pause packet transmission is enabled by setting the MFFCNT.IVPAUSE bit. Time required to transmit one packet = Time required to transmit 512 bits (circuit size) = 128 TXCLK cycles Default value: About 168 ms when the data rate is 100 Mbps, or about 1.68 seconds when the data rate is 10 Mbps
15 to 0	PAUSETM_MAX[15:0]	Pause control timer value of MAX pause packet These bits specify the value of TPTV[15:0] when a pause control request is issued to the MAC. Time required to transmit one packet = Time required to transmit 512 bits (circuit size) = 128 TXCLK cycles Default value: About 336 ms when the data rate is 100 Mbps, or about 3.36 seconds when the data rate is 10 Mbps

(5) ETHAnRXERSEL - Receive error selection register

This register is used to specify whether to accept or discard the packet if a reception error occurs.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0220_H

Initial value $0000\ 0001_H$. This register is initialized by any reset.

Caution Be sure to set bits 25 to 23 and 15 to 2 to “0”.

31	30	29	28	27	26	25	24
RLENE	VLAN	USOP	RPCF	RCFR	DBNB	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R
23	22	21	20	19	18	17	16
0	RLOR	RLER	RCRCE	RXER	CEPS	REPS	PAIG
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	TXRX	DVCF
R	R	R	R	R	R	R/W	R/W

Table 23-79 ETHAnRXERSEL register contents (1/2)

Bit position	Bit name	Function
31	RLENE	Specifies whether to accept a packet that contains a packet length error. 0: Accepts the packet. 1: Discards the packet.
30	VLAN	Specifies whether to accept a VLAN packet. 0: Accepts the packet. 1: Discards the packet.
29	USOP	Specifies whether to accept an undefined opcode control packet. 0: Accepts the packet. 1: Discards the packet.
28	RPCF	Specifies whether to accept a pause control packet. 0: Accepts the packet. 1: Discards the packet.
27	RCFR	Specifies whether to accept a control packet. 0: Accepts the packet. 1: Discards the packet.
26	DBNB	Specifies whether to accept a packet that contains dribble nibble. 0: Accepts the packet. 1: Discards the packet.
22	RLOR	Specifies whether to accept a packet that has a length field exceeding 1,500 bytes. 0: Accepts the packet. 1: Discards the packet.

Table 23-79 ETHAnRXERSEL register contents (2/2)

Bit position	Bit name	Function
21	RLER	Specifies whether to accept a packet whose length field does not match the data field length. 0: Accepts the packet. 1: Discards the packet.
20	RCRCE	Specifies whether to accept a packet in which a CRC error has occurred. 0: Accepts the packet. 1: Discards the packet.
19	RXER	Specifies whether to accept a packet in which RXER has been detected. 0: Accepts the packet. 1: Discards the packet.
18	CEPS	Specifies whether to accept a packet in which a false carrier has been detected. 0: Accepts the packet. 1: Discards the packet.
17	REPS	Specifies whether to accept a packet in which the total size of the preamble and SFD or the data field size is one nibble. 0: Accepts the packet. 1: Discards the packet.
16	PAIG	Specifies whether to accept a packet for which one of the following events occurred after the previous packet was received. • A carrier longer than 6,072 nibbles (3,036 bytes) has been detected. • The next packet with IFG + preamble + SFD has been received before the time required to transmit 80 bits has elapsed after a packet has been received. • An illegal preamble or SFD has been received while the pure preamble data check is enabled (the ETHAnMACC1.PUREP bit is set). 0: Accepts the packet. 1: Discards the packet.
1	TXRX	Specifies whether to accept a packet in which a collision has been detected by the MAC. 0: Accepts the packet. 1: Discards the packet.
0	DVCF	Specifies whether to accept a packet received by the MAC that is judged to be a valid control packet. 0: Accepts the packet. 1: Discards the packet.

(6) ETHAnTXSTMONI1 - Transmission status monitor 1 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0230_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	CSE	TBP	TPP	TPCF	TCFR
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
TTBC15	TTBC14	TTBC13	TTBC12	TTBC11	TTBC10	TTBC9	TTBC8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
TTBC7	TTBC6	TTBC5	TTBC4	TTBC3	TTBC2	TTBC1	TTBC0
R	R	R	R	R	R	R	R

Table 23-80 ETHAnTXSTMONI1 register contents

Bit position	Bit name	Function
20	CSE	Detection of carrier loss during transmission
19	TBP	Occurrence of a collision due to the back pressure function after the previous transmission[a]
18	TPP	End of transmission of a transmit packet requested during pausing[b]
17	TPCF	Transmission of a pause control packet
16	TCFR	Transmission of a control packet
15 to 0	TTBC[15:0]	Number of bytes of transmitted data, including packets in which collisions occurred

a) This bit is set if a collision has occurred between when the transmission status was previously updated and when the status is updated this time.

b) This bit is not set if the packet requested during pausing is a control frame.

(7) ETHAnTXSTMONI2 - Transmission status monitor 2 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0234_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TUDR	TGNT	LCOL	ECOL	TEDFR	TDFR	TBRO	TMUL
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
TDONE	TFLOR	TFLER	TCRCE	TCBC3	TCBC2	TCBC1	TCBC0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
TBYT15	TBYT14	TBYT13	TBYT12	TBYT11	TBYT10	TBYT9	TBYT8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
TBYT7	TBYT6	TBYT5	TBYT4	TBYT3	TBYT2	TBYT1	TBYT0
R	R	R	R	R	R	R	R

Table 23-81 ETHAnTXSTMONI2 register contents

Bit position	Bit name	Function
31	TUDR	Detection of a transmit packet underrun[a]
30	TGNT	Transmission of a packet longer than the value specified by ETHAnLMAX[b]
29	LCOL	Occurrence of a late collision
28	ECOL	Occurrence of collisions exceeding the specified maximum number
27	TEDFR	Detection of an excessive transmission delay
26	TDFR	Occurrence of a transmission delay
25	TBRO	Transmission of a broadcast packet
24	TMUL	Transmission of a multicast packet
23	TDONE	End of transmission[c]
22	TFLOR	Detection of a length field greater than 1,500 bytes[d]
21	TFLER	Detection of a packet whose length field does not match the data field length[d, e]
20	TCRCE	Occurrence of a CRC error when CRC automatic appending mode was disabled
19:16	TCBC[3:0]	Number of times retransmission occurred due to collisions[f]
15 to 0	TBYT[15:0]	Transmit packet length (number of bytes) when transmission finished normally[f]

a) This bit is set only if no collision has occurred.
b) This bit is set only if ETHAnMACC1.HUGEN is 0.
c) This bit is not set if transmission has been aborted.
d) This bit is not set if ETHAnMACC1.FLCHT is 0.
e) If the length field exceeds 1,500 bytes, TFLOR is set and TFLER is not.
f) This value is not correct if transmission has been aborted.

(8) ETHAnTXFINF1 - Transmission status 1 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0238_H

Initial value $0000\ 0800_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	TPCNT8
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
TPCNT7	TPCNT6	TPCNT5	TPCNT4	TPCNT3	TPCNT2	TPCNT1	TPCNT0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	TREMAIN11	TREMAIN10	TREMAIN9	TREMAIN8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
TREMAIN7	TREMAIN6	TREMAIN5	TREMAIN4	TREMAIN3	TREMAIN2	TREMAIN1	TREMAIN0
R	R	R	R	R	R	R	R

Table 23-82 ETHAnTXFINF1 register contents

Bit position	Bit name	Function
24:16	TPCNT[8:0]	Number of packets in the transmit FIFO These bits indicate the number of packets (start flag to end flag) that exist in the transmit FIFO. The value is incremented when one packet has been written by the system. The value is decremented when one packet has been read by the MAC (upon completion or halting of transmission).
11:0	TREMAIN[11:0]	These bits indicate the remaining capacity of the transmit FIFO (in bytes). Bits 1 and 0 are always 00 because the data in the transmit FIFO are aligned in 32-bit (4-byte) units.

(9) ETHAnTXFINF2 - Transmission status 2 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + $023C_H$

Initial value $0000\ 0001_H$. This register is initialized by any reset.

Caution Before rewriting a mode register related to transmission, be sure to confirm that the ETHAnTXFINF2.TXSTOP bit is 1.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	TXSTOP
R	R	R	R	R	R	R	R

Table 23-83 ETHAnTXFINF2 register contents

Bit position	Bit name	Function
0	TXSTOP	This bit is set if no data is in the transmit FIFO while transmission is stopped (by clearing ETHAnMFFCONT.TXEN). Before rewriting a mode setting register related to transmission, be sure to confirm that this bit is 1. 0: The transmit FIFO is operating. 1: The transmit FIFO is stopped.

(10) ETHAnRXSTMONI - Reception status monitor register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0240_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RLENE	VLAN	USOP	RPCF	RCFR	DBNB	RBRO	RMUL
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
RXOK	RLOR	RLER	RCRCE	RXER	CEPS	REPS	PAIG
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
RBYT15	RBYT14	RBYT13	RBYT12	RBYT11	RBYT10	RBYT9	RBYT8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
RBYT7	RBYT6	RBYT5	RBYT4	RBYT3	RBYT2	RBYT1	RBYT0
R	R	R	R	R	R	R	R

Table 23-84 ETHAnRXSTMONI register contents

Bit position	Bit name	Function
31	RLENE	Occurrence of a receive packet length error This bit is set if the received packet size is less than 64 bytes or greater than 1,518 bytes (or less than 64 bytes or greater than 1,522 bytes for a VLAN packet).
30	VLAN	Reception of a VLAN packet This bit is set if a packet whose TPID field matches ETHAnVLTP is received.[a]
29	USOP	Reception of an undefined opcode control packet[b]
28	RPCF	Reception of a pause control packet[b]
27	RCFR	Reception of a control packet[b]
26	DBNB	Reception of a packet containing dribble nibble
25	RBRO	Reception of a broadcast packet
24	RMUL	Reception of a multicast packet
23	RXOK	End of reception[a]
22	RLOR	Reception of a packet that has a length field exceeding 1,500 bytes[c]
21	RLER	Detection of a packet whose length field does not match the data field length[c, d]
20	RCRCE	Occurrence of a CRC error
19	RXER	Detection of RXER
18	CEPS	Detection of a false carrier [e]
17	REPS	Reception of a packet in which the total size of the preamble and SFD or the data field size is one nibble[e, f]
16	PAIG	This bit is set if one of the following events occurred after the previous packet was received.[e] • A carrier longer than 6,072 nibbles (3,036 bytes) has been detected. • The next packet with IFG + preamble + SFD has been received before the time required to transmit 80 bits has elapsed after a packet has been received.[f] • An illegal preamble or SFD has been received while the pure preamble data check is enabled (the ETHAnMACC1.PUREP bit is set).[f]
15 to 0	RBYT[15:0]	Number of received bytes

a) This bit is not set if a CRC error or RXER has occurred.
b) This bit is not set if a CRC error has occurred.
c) This bit is not set if ETHAnMACC1.FLCHT is 0.
d) If the length field exceeds 1,500 bytes, RLOR is set and RLER is not.
e) The bit is set if a false carrier is detected between when the reception status was updated previously and when it is updated this time.
f) A packet in which this event has occurred is ignored and not transferred to the upstream system.
The ETHAnRXSTMONI register is updated when a DMA transfer of the receive packet has finished. (This register indicates the status of the packet transferred by DMA.) The ETHAnRXFINF1 register is also updated at the same time.

The ETHAnRXSTATUS register is updated when a DMA transfer of the receive packet has finished. The reception status register 1 (ETHAnRXFINF1) is also updated at the same time.

(11) ETHAnRXFINF1 - Reception status 1 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0244_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
RPLEN15	RPLEN14	RPLEN13	RPLEN12	RPLEN11	RPLEN10	RPLEN9	RPLEN8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
RPLEN7	RPLEN6	RPLEN5	RPLEN4	RPLEN3	RPLEN2	RPLEN1	RPLEN0
R	R	R	R	R	R	R	R

Table 23-85 ETHAnRXFINF1 register contents

Bit position	Bit name	Function
15 to 0	RPLEN[15:0]	These bits indicate the receive packet length in bytes. The Ethernet controller uses the value of RPLEN for the size field when writing back the receive descriptor.

(12) ETHAnRXFINF2 - Reception status 2 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0248_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	RPCNT8
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
RPCNT7	RPCNT6	RPCNT5	RPCNT4	RPCNT3	RPCNT2	RPCNT1	RPCNT0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	RREMAIN11	RREMAIN10	RREMAIN9	RREMAIN8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
RREMAIN7	RREMAIN6	RREMAIN5	RREMAIN4	RREMAIN3	RREMAIN2	RREMAIN1	RREMAIN0
R	R	R	R	R	R	R	R

Table 23-86 ETHAnRXFINF2 register contents

Bit position	Bit name	Function
24:16	RPCNT[8:0]	Number of packets in the receive FIFO These bits indicate the number of packets (start flag to end flag) that exist in the receive FIFO. The value is incremented when one packet has been written by the MAC. (A packet that is discarded upon being received does not increment this value.) If a packet has been read from the DMAC for the Ethernet controller or if the packet is discarded, this value is decremented after the internal operation to discard the packet finishes.
11:0	RREMAIN[11:0]	These bits indicates the remaining receive FIFO capacity (in bytes). Bits 1 and 0 are always 00 because the data in the receive FIFO is aligned in 32-bit (4-byte) units.

(13) ETHAnRXFINF3 - Reception status 3 register

This register indicates the status of the receive FIFO while reception is stopped.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + $024C_H$

Initial value $0000\ 0001_H$. This register is initialized by any reset.

Caution Before rewriting a mode setting register related to reception or flow control, be sure to confirm that the ETHAnRXFINF3.RXSTOP bit is 1.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	RXSTOP
R	R	R	R	R	R	R	R

Table 23-87 ETHAnRXFINF3 register contents

Bit position	Bit name	Function
0	RXSTOP	This bit is set if no data is in the receive FIFO while reception is stopped (by clearing ETHAnMFFCONT.RXEN). Before rewriting a mode setting register related to reception or flow control, confirm that the RXSTOP bit is 1. 0: Receive FIFO is operating. 1: Receive FIFO is stopped.

(14) ETHAnFSTATUS - FIFO status interrupt register

If an interrupt source that is not masked by the ETHA0FSTATMK register is generated, the INTETMFS interrupt (FIFO status interrupt) is generated. The INTETMFS interrupt signal is asserted until all bits in this register are cleared.

If an interrupt source masked by the ETHA0FSTATMK register is generated, the corresponding bit in this register is set. All the bits of the ETHAnFSTATUS register are cleared when the register is read.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0250_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution The FIFO status interrupt status register is cleared when it is read. It is recommended to copy interrupt sources to variables so that multiple interrupt sources generated concurrently can be detected.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	TACOF
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	RACOF
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	TSUP	TFNRTY	TFWE	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
RFFE	RSUP	0	RFWE	RFOF	0	RFFLW	RFZP
R	R	R	R	R	R	R	R

Table 23-88 ETHAnFSTATUS register contents

Bit position	Bit name	Function
24	TACOF	This bit is set when the ETHAnTXABTCNT register (TX abort counter) overflows.
16	RACOF	This bit is set when the ETHAnRXABTCNT register (RX abort counter) overflows.
12	TSUP	TX status update This bit is set when the transmission status is updated in the ETHAnTXSTMONI1 or ETHAnTXSTMONI2 register.
11	TFNRTY	Transmit FIFO abort (transmit FIFO no retry) This bit is set if transmission fails and the data in the transmit FIFO is discarded. In this case, ETHAnTXABTCNT is incremented.
10	TFWE	This bit is set if a transmit FIFO write error has occurred.
7	RFFE	Receive FIFO flag error This bit is set if handshaking is not correctly performed when receive data is written from the MAC to the receive FIFO. The receive packet and reception status are invalid but reception is not canceled. • If the reception status is updated before all receive data is stored in the receive FIFO, the end of the packet is assumed as soon as the reception status has been updated. • If the reception status is not updated after all receive data has been stored in the receive FIFO, the reception status is assumed to be all 0.
6	RSUP	This bit is set when the reception status monitor register (ETHAnRXSTMONI) is updated. A valid value can be read from the ETHAnRXSTMONI or ETHAnRXFINF1 register when this bit is 1.
4	RFWE	Receive FIFO write error This bit is set if a packet whose size is less than 32 bits (4 bytes) has been received and could not be written to the receive FIFO.
3	RFOF	This bit is set if the receive FIFO overflows.
1	RFFLW	This bit is set if the data in the receive FIFO reaches or exceeds the value specified by the ETHA0FLOWTH.FLOWTHR bits.
0	RFZP	This bit is set if the data in the receive FIFO reaches or exceeds the value specified by the ETHA0FLOWTH.ZPTHR bits.

(15) ETHA0FSTATMK - FIFO status interrupt mask register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0254_H

Initial value $0101\ 1FFF_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 25, 23 to 17, and 15 to 13 to “0”, and set bits 9, 8, 5, and 2 to “1”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	TACOF
R	R	R	R	R	R	R	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	RACOF
R	R	R	R	R	R	R	R/W
15	14	13	12	11	10	9	8
0	0	0	TSUP	TFNRTY	TFWE	1	1
R	R	R	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RFFE	RSUP	1	RFWE	RFOF	1	RFFLW	RFZP
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-89 ETHA0FSTATMK register contents

Bit position	Bit name	Function
24	TACOF	0: Does not mask interrupt. 1: Masks interrupt.
16	RACOF	0: Does not mask interrupt. 1: Masks interrupt.
12	TSUP	0: Does not mask interrupt. 1: Masks interrupt.
11	TFNRTY	0: Does not mask interrupt. 1: Masks interrupt.
10	TFWE	0: Does not mask interrupt. 1: Masks interrupt.
7	RFFE	0: Does not mask interrupt. 1: Masks interrupt.
6	RSUP	0: Does not mask interrupt. 1: Masks interrupt.
4	RFWE	0: Does not mask interrupt. 1: Masks interrupt.
3	RFOF	0: Does not mask interrupt. 1: Masks interrupt.
1	RFFLW	0: Does not mask interrupt. 1: Masks interrupt.
0	RFZP	0: Does not mask interrupt. 1: Masks interrupt.

(16) ETHAnTXSTATUS - Transmission status interrupt register

This register stores the cumulative result of the transmission status. If an interrupt that is not masked by the ETHAnTXSTATMK register is generated, the INTETMTS interrupt (transmission status interrupt) is generated. The INTETMTS interrupt signal is asserted until all bits in this register are cleared.

If an interrupt source masked by the ETHAnTXSTATMK register is generated, the corresponding bit in this register is set. All the bits of the ETHAnTXSTATUS register are cleared when the register is read.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0258_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution The transmission status interrupt register is cleared when it is read. It is recommended to copy interrupt sources to variables so that multiple interrupt sources generated concurrently can be detected.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	TAB
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
TGNT	LCOL	ECOL	TEDFR	TDFR	TFLOR	TFLER	TCRCE
R	R	R	R	R	R	R	R

Table 23-90 ETHAnTXSTATUS register contents

Bit position	Bit name	Function
16	TAB	This bit is set if transmission has been aborted.
7	TGNT	This bit is set if a packet longer than the value specified by ETHAnLMAX has been transmitted (TAB source). This bit is not set if ETHAnMACC1.HUGEN is 1.
6	LCOL	This bit is set if a late collision has been detected (TAB source).
5	ECOL	This bit is set if collisions have occurred exceeding the specified maximum number (TAB source).
4	TEDFR	This bit is set if an excessive transmission delay has been detected (TAB source).
3	TDFR	This bit is set if a transmission delay has occurred.
2	TFLOR	This bit is set if a packet whose length field is greater than 1,500 bytes is detected. This bit is also set when a VLAN packet pause control frame is transmitted. This bit is not set if ETHAnMACC1.FLCHT is 0.
1	TFLER	This bit is set if a packet whose length field does not match the data field length is detected. This bit is not set if ETHAnMACC1.FLCHT is 0. If the length field exceeds 1,500 bytes, TFLOR is set and TFLER is not.
0	TCRCE	This bit is set if a CRC error occurred. This bit is set if transmission is performed with the CRC automatic appending mode disabled (the ETHAnMACC1.PADEN and ETHAnMACC1.CRCEN bits are 0).

(17) ETHAnTXSTATMK - Transmission status interrupt mask register

This register is used to mask the transmission status interrupt (INTCTS).

If an interrupt source that is not masked by this register is generated, the INTCTS interrupt is generated. The INTCTS interrupt signal is asserted until all the related interrupt sources are cleared. If an interrupt source masked by the ETHAnTXSTATMK register is generated, the corresponding bit in the ETHAnTXSTATUS register is set.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $025C_H$

Initial value $0001\ 01FF_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 17 and 15 to 8 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	TAB
R	R	R	R	R	R	R	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
TGNT	LCOL	ECOL	TEDFR	TDFR	TFLOR	TFLER	TCRCE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-91 ETHAnTXSTATMK register contents

Bit position	Bit name	Function
16	TAB	0: Does not mask interrupt. 1: Masks interrupt.
7	TGNT	0: Does not mask interrupt. 1: Masks interrupt.
6	LCOL	0: Does not mask interrupt. 1: Masks interrupt.
5	ECOL	0: Does not mask interrupt. 1: Masks interrupt.
4	TEDFR	0: Does not mask interrupt. 1: Masks interrupt.
3	TDFR	0: Does not mask interrupt. 1: Masks interrupt.
2	TFLOR	0: Does not mask interrupt. 1: Masks interrupt.
1	TFLER	0: Does not mask interrupt. 1: Masks interrupt.
0	TCRCE	0: Does not mask interrupt. 1: Masks interrupt.

(18) ETHAnRXSTATUS - Reception status interrupt register

This register stores the cumulative result of the reception status. If an interrupt source that is not masked by the ETHAnRXSTATMK register is generated, the INTETMRS interrupt is generated. The INTETMRS interrupt signal is asserted until all bits in this register are cleared.

If an interrupt source masked by the ETHAnRXSTATMK register is generated, the corresponding bit in this register is set.

This register is not affected by the setting of ETHAnRXERSEL (receive error selection register).

All the bits of the ETHAnRXSTATUS register are cleared when the register is read.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0260_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution The reception status interrupt status register is cleared when it is read. It is recommended to copy interrupt sources to variables so that multiple interrupt sources generated concurrently can be detected.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	RLENE	VLAN	USOP	RPCF	RCFR	DBNB	RLOR
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
RLER	RCRCE	RXER	CEPS	REPS	PAIG	TXRX	DVCF
R	R	R	R	R	R	R	R

Table 23-92 ETHAnRXSTATUS register contents

Bit position	Bit name	Function
14	RLENE	Occurrence of a receive packet length error This bit is set if the received packet size is less than 64 bytes or greater than 1,518 bytes (or less than 64 bytes or greater than 1,522 bytes for a VLAN packet).
13	VLAN	Reception of a VLAN packet This bit is set if a packet whose TPID field matches ETHAnVLTP.[a]
12	USOP	Reception of an undefined opcode control packet[b]
11	RPCF	Reception of a pause control packet[b]
10	RCFR	Reception of a control packet[b]
9	DBNB	Reception of a packet containing dribble nibble
8	RLOR	Reception of a packet that has a length field exceeding 1,500 bytes[c]
7	RLER	Detection of a packet whose length field does not match the data field length[c, d]
6	RCRCE	Occurrence of a receive CRC error
5	RXER	Detection of RXER
4	CEPS	Detection of a false carrier[e]
3	REPS	Reception of a packet in which the total size of the preamble and SFD or the data field size is one nibble[e, f]
2	PAIG	This bit is set if one of the following events occurred after the previous packet was received.[e] • A carrier longer than 6,072 nibbles (3,036 bytes) has been detected. • The next packet with IFG + preamble + SFD has been received before the time required to transmit 80 bits has elapsed after a packet has been received. • An illegal preamble or SFD has been received while the pure preamble data check is enabled (the ETHAnMACC1.PUREP bit is set).
1	TXRX	This bit is set if transmission starts (a collision occurs) during half-duplex reception (immediately after starting reception).
0	DVCF	This bit is set if the received packet is a valid control packet (that does not contain an error).

a) This bit is not set if a CRC error or RXER has occurred.
b) This bit is not set if a CRC error has occurred.
c) This bit is not set if ETHAnMACC1.FLCHT is 0.
d) If the length field exceeds 1,500 bytes, RLOR is set and RLER is not.
e) This bit indicates that this event occurred between when the reception status was updated previously and when it is updated this time.
f) A packet in which this event has occurred is ignored and not transferred to the upstream system.

(19) ETHAnRXSTATMK - Reception status interrupt mask register

This register is used to mask the reception status interrupt (INTETMRS).

If an interrupt source that is not masked by this register is generated, INTETMRS is generated. The INTETMRS interrupt signal is asserted until all the related interrupt sources are cleared. If an interrupt source masked by this register is generated, the corresponding bit in the ETHAnRXSTATUS register is set.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0264_H

Initial value $0000\ 07FF_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 15 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	RLENE	VLAN	USOP	RPCF	RCFR	DBNB	RLOR
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RLER	RCRCE	RXER	CEPS	REPS	PAIG	TXRX	DVCF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-93 ETHAnRXSTATMK register contents

Bit position	Bit name	Function
14	RLENE	0: Does not mask interrupt. 1: Masks interrupt.
13	VLAN	0: Does not mask interrupt. 1: Masks interrupt.
12	USOP	0: Does not mask interrupt. 1: Masks interrupt.
11	RPCF	0: Does not mask interrupt. 1: Masks interrupt.
10	RCFR	0: Does not mask interrupt. 1: Masks interrupt.
9	DBNB	0: Does not mask interrupt. 1: Masks interrupt.
8	RLOR	0: Does not mask interrupt. 1: Masks interrupt.
7	RLER	0: Does not mask interrupt. 1: Masks interrupt.
6	RCRCE	0: Does not mask interrupt. 1: Masks interrupt.
5	RXER	0: Does not mask interrupt. 1: Masks interrupt.
4	CEPS	0: Does not mask interrupt. 1: Masks interrupt.
3	REPS	0: Does not mask interrupt. 1: Masks interrupt.
2	PAIG	0: Does not mask interrupt. 1: Masks interrupt.
1	TXRX	0: Does not mask interrupt. 1: Masks interrupt.
0	DVCF	0: Does not mask interrupt. 1: Masks interrupt.

(20) ETHAnTXABTCNT - TX abort counter

This is a transmission abort counter. It counts the number of packets that have resulted in a MAC transmission error (including an underrun).

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0270_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
TABCNT15	TABCNT14	TABCNT13	TABCNT12	TABCNT11	TABCNT10	TABCNT9	TABCNT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TABCNT7	TABCNT6	TABCNT5	TABCNT4	TABCNT3	TABCNT2	TABCNT1	TABCNT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-94 ETHAnTXABTCNT register contents

	Bit position	Bit name	Function
<R>	15 to 0	TABCNT[15:0]	Transmission abort count These bits count the number of packets that have resulted in a MAC transmission error (including an underrun). This counter is not incremented when ETHAnMFFCONT.TABT is set to retransmit an aborted packet. Packets are counted after 68 bytes have been transferred because they are not retransmitted by a retry request (usually, the MAC does not issue a retry request after 64 bytes have been transferred). If the count value overflows, the value of these bits returns to 0 and the ETHAnFSTATUS.TACOF bit is set. These bits are not cleared by resetting the transmission circuit (by setting TFRST and TFFLSH).

(21) ETHAnRXABTCNT - RX abort counter

This is a reception abort counter. These bits count the number of receive packets that have been discarded because of the status of the receive packet, the status of the receive FIFO, address filtering by the MAC, and reception of a control packet.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0274_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
TABCNT15	TABCNT14	TABCNT13	TABCNT12	TABCNT11	TABCNT10	TABCNT9	TABCNT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TABCNT7	TABCNT6	TABCNT5	TABCNT4	TABCNT3	TABCNT2	TABCNT1	TABCNT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-95 ETHAnRXABTCNT register contents

Bit position	Bit name	Function
15 to 0	TABCNT[15:0]	Reception abort count These bits count the number of receive packets that have been discarded because of the status of the receive packet, the status of the receive FIFO, address filtering by the MAC, and reception of a control packet. If the count value overflows, the value of these bits returns to 0 and the ETHAnFSTATUS.RACOF bit is set. These bits are not cleared by resetting the reception circuit (by setting RFRST and RFFLSH).

23.4.4 DMAC control registers for Ethernet controller

(1) ETHAnMODE - Core function control register

This register is used to analyze the reception start descriptor and transmission start descriptor.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0300_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 19 and 16 to 0 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	RXS	TXS	0
R	R	R	R	R	R/W	R/W	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

Table 23-96 ETHAnMODE register contents

Bit position	Bit name	Function
18	RXS	Reception enable bit The RXS bit is used to start analyzing the reception start descriptor. This bit is automatically cleared after 1 is written to it.
17	TXS	Transmission enable bit The TXS bit is used to start analyzing the transmission start descriptor. This bit is automatically cleared after 1 is written to it.

(2) ETHAnINTMS - Interrupt control register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0304_H

Initial value $0700\ 0700_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 28, 23 to 20, 15 to 11, and 7 to 3 to “0”.

Note The RBEI, RECI, RXI, TBEI, TECI, and TXI bits of the ETHAnINTMS register are cleared when they are read.

31	30	29	28	27	26	25	24
0	0	0	0	0	RBEMSK	RECMSK	RXMSK
R	R	R	R	R	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	RBEI	RECI	RXI
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	TBEMSK	TECMSK	TXMSK
R	R	R	R	R	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	TBEI	TECI	TXI
R	R	R	R	R	R	R	R

Table 23-97 ETHAnINTMS register contents (1/2)

Bit position	Bit name	Function
27	RUSMSK	This bit specifies whether to mask the RUPI interrupt indicated by bit 19. 0: Does not mask interrupt. 1: Masks interrupt.
26	RBEMSK	This bit specifies whether to mask the RBEI interrupt indicated by bit 18. 0: Does not mask interrupt. 1: Masks interrupt.
25	RECMSK	This bit specifies whether to mask the RECI interrupt indicated by bit 17. 0: Does not mask interrupt. 1: Masks interrupt.
24	RXMSK	This bit specifies whether to mask the RXI interrupt indicated by bit 16. 0: Does not mask interrupt. 1: Masks interrupt.
19	RUPI	This bit indicates whether a pause interrupt specified by the “U” (used) bit of a receive descriptor has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
18	RBEI	This bit indicates whether a receive data buffer access error interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
17	RECI	This bit indicates whether a reception (DMA) end of chain interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
16	RXI	This bit indicates whether a packet reception (DMA) completion interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
11	TUSMSK	This bit specifies whether to mask the TUPI interrupt indicated by bit 3. 0: Does not mask interrupt. 1: Masks interrupt.
10	TBEMSK	This bit specifies whether to mask the TBEI interrupt indicated by bit 2. 0: Does not mask interrupt. 1: Masks interrupt.
9	TECMSK	This bit specifies whether to mask the TECI interrupt indicated by bit 1. 0: Does not mask interrupt. 1: Masks interrupt.
8	TXMSK	This bit specifies whether to mask the TXI interrupt indicated by bit 0. 0: Does not mask interrupt. 1: Masks interrupt.
3	TUPI	This bit indicates whether a pause interrupt specified by the “U” (used) bit of a transmit descriptor has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.

Table 23-97 ETHAnINTMS register contents (2/2)

Bit position	Bit name	Function
2	TBEI	This bit indicates whether a transmit data buffer access error interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
1	TECI	This bit indicates whether a transmission (DMA) end of chain interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
0	TXI	This bit indicates whether a packet transmission (DMA) completion interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.

(3) ETHAnTRANSCTL - Transfer control register

This register is used to control transfer by the DMAC for the Ethernet controller.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0308_H

Initial value $0003\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 26, 23 to 18, and 15 to 1 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	RXEN_STA	TXEN_STA
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	RXEN	TXEN
R	R	R	R	R	R	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	RXCHKSMEN
R	R	R	R	R	R	R	R/W

Table 23-98 ETHAnTRANSCTL register contents

Bit position	Bit name	Function
25	RXEN_STA	This bit indicates the reception status. 0: Reception is not in progress (IDLE status). 1: Reception is in progress. Reception stops if an RBEI or RECI interrupt, which is controlled by the interrupt register (ETHAnINTMS), is generated. At this time, this bit is cleared.
24	TXEN_STA	This bit indicates the transmission status. 0: Transmission is not in progress (IDLE status). 1: Transmission is in progress. Transmission stops if a TBEI or TECI interrupt, which is controlled by the interrupt register (ETHAnINTMS), is generated. At this time, this bit is cleared.
17	RXEN	This bit specifies whether to enable reception. 0: Disables reception (DMA reception stops.) 1: Enables reception.
16	TXEN	This bit specifies whether to enable transmission. 0: Disables transmission (DMA transmission stops). 1: Enables transmission.
0	RXCHKSMEN	This bit enables or disables the receive checksum appending function. 0: Disables the receive check sum appending function. 1: Enables the receive check sum appending function.

(4) ETHAnSFTRST - Software reset setting register

This register is used to execute a software reset for the DMAC for the Ethernet controller.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $030C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 1 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	SFTRST
R	R	R	R	R	R	R	R/W

Table 23-99 ETHAnSFTRST register contents

Bit position	Bit name	Function
0	SFTRST	Software reset When this bit is set, the DMAC for the Ethernet controller is reset. The receive checksum unit is also reset. This bit is automatically cleared after 1 is written to it.

(5) ETHAnDMACM - DMA mode control register

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0310_H

Initial value $0000\ 0010_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 11, 7 to 5, and 3 to 0 to “0”, and bit 4 to “1”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	BURST2	BURST1	BURST0
R	R	R	R	R	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

Table 23-100 ETHAnDMACM register contents

Bit position	Bit name	Function
10:8	BURST[2:0]	These bits specify the type of burst transfer.

BURST2	BURST1	BURST0	Type	Operation
0	0	0	SINGLE	Single transfer mode
0	1	1	INCR4	4-beat incremental burst transfer mode
1	0	1	INCR8	8-beat incremental burst transfer mode
1	1	1	INCR16	16-beat incremental burst transfer mode
Other than above			Setting prohibited	

(6) ETHAnRXDP - Receive descriptor pointer register

This register is used to specify the pointer position of the receive descriptor of the DMAC for the Ethernet controller.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0320_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
RXDP31	RXDP30	RXDP29	RXDP28	RXDP27	RXDP26	RXDP25	RXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
RXDP23	RXDP22	RXDP21	RXDP20	RXDP19	RXDP18	RXDP17	RXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
RXDP15	RXDP14	RXDP13	RXDP12	RXDP11	RXDP10	RXDP9	RXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
RXDP7	RXDP6	RXDP5	RXDP4	RXDP3	RXDP2	RXDP1	RXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 23-101 ETHAnRXDP register contents

Bit position	Bit name	Function
31 to 0	RXDP[31:0]	These bits specify the pointer position of the receive descriptor. Specify the first address of the receive descriptor chain. Bits 1 and 0 are fixed to 00.

(7) ETHAnLSTRXDP - Last receive descriptor pointer register

This register indicates the last receive descriptor address of the DMAC for the Ethernet controller.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0320_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
LSTRXDP31	LSTRXDP30	LSTRXDP29	LSTRXDP28	LSTRXDP27	LSTRXDP26	LSTRXDP25	LSTRXDP24
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
LSTRXDP23	LSTRXDP22	LSTRXDP21	LSTRXDP20	LSTRXDP19	LSTRXDP18	LSTRXDP17	LSTRXDP16
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
LSTRXDP15	LSTRXDP14	LSTRXDP13	LSTRXDP12	LSTRXDP11	LSTRXDP10	LSTRXDP9	LSTRXDP8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
LSTRXDP7	LSTRXDP6	LSTRXDP5	LSTRXDP4	LSTRXDP3	LSTRXDP2	LSTRXDP1	LSTRXDP0
R	R	R	R	R	R	R	R

Table 23-102 ETHAnLSTRXDP register contents

Bit position	Bit name	Function
31 to 0	LSTRXDP[31:0]	These bits indicate the last receive descriptor pointer address. They hold the address of the last accessed receive descriptor. Bits 1 and 0 are fixed to 00.

(8) ETHAnTXDP - Transmit descriptor pointer register

This register is used to specify the pointer position of the transmit descriptor of the DMAC for the Ethernet controller.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + 0328_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TXDP31	TXDP30	TXDP29	TXDP28	TXDP27	TXDP26	TXDP25	TXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TXDP23	TXDP22	TXDP21	TXDP20	TXDP19	TXDP18	TXDP17	TXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TXDP15	TXDP14	TXDP13	TXDP12	TXDP11	TXDP10	TXDP9	TXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TXDP7	TXDP6	TXDP5	TXDP4	TXDP3	TXDP2	TXDP1	TXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 23-103 ETHAnTXDP register contents

Bit position	Bit name	Function
31 to 0	TXDP[31:0]	These bits specify the pointer position of the transmit descriptor. Specify the first address of the transmit descriptor chain. Bits 1 and 0 are fixed to 00.

(9) ETHAnLSTTXDP - Last transmit descriptor pointer register

This register indicates the last transmit descriptor address of the DMAC for the Ethernet controller.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + $032C_H$

Initial value $FFFF\ FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
LSTTXDP31	LSTTXDP30	LSTTXDP29	LSTTXDP28	LSTTXDP27	LSTTXDP26	LSTTXDP25	LSTTXDP24
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
LSTTXDP23	LSTTXDP22	LSTTXDP21	LSTTXDP20	LSTTXDP19	LSTTXDP18	LSTTXDP17	LSTTXDP16
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
LSTTXDP15	LSTTXDP14	LSTTXDP13	LSTTXDP12	LSTTXDP11	LSTTXDP10	LSTTXDP9	LSTTXDP8
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
LSTTXDP7	LSTTXDP6	LSTTXDP5	LSTTXDP4	LSTTXDP3	LSTTXDP2	LSTTXDP1	LSTTXDP0
R	R	R	R	R	R	R	R

Table 23-104 ETHAnLSTTXDP register contents

Bit position	Bit name	Function
31 to 0	LSTTXDP[31:0]	These bits indicate the last transmit descriptor pointer address. They hold the address of the last accessed transmit descriptor. Bits 1 and 0 are fixed to 00.

23.4.5 Control registers of DMAC for transmit checksum

(1) ETHAnCMODE - Transmit checksum unit function setting register

This register is used to analyze the reception start descriptor and transmission start descriptor.

Access This register can be read or written in 32-bit units.

Address <ETHAnC_base> + 0300_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	TCH_RXS	TCH_TXS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 23-105 ETHAnCMODE register contents

Bit position	Bit name	Function
18	TCH_RXS	The TCH_RXS bit is used to start analyzing the reception start descriptor. This bit is cleared immediately after 1 is written to it.
17	TCH_TXS	The TCH_TXS bit is used to start analyzing the transmission start descriptor. This bit is cleared immediately after 1 is written to it.

(2) ETHAnCINTMS - Transmit checksum interrupt register

This register indicates the status of and masks the INTSCRXTCH and INTSCTXTCH interrupts of the DMAC for the transmit checksum.

Access This register can be read or written in 32-bit units. Bits 18 to 16 and 2 to 0 can only be read, however.

Address <ETHAnC_base> + 0304_H

Initial value $0700\ 0700_H$. This register is initialized by any reset.

Caution The TCH_RBEI, TCH_RECI, TCH_RXI, TCH_TEBI, TCH_TECI, and TCH_TXI bits of the transmit checksum interrupt register are cleared when they are read. It is recommended to copy interrupt sources to variables so that multiple interrupt sources generated concurrently can be detected.

31	30	29	28	27	26	25	24
0	0	0	0	0	TCH_RBEMSK	TCH_RECMSK	TCH_RXMSK
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	TCH_RBEI	TCH_RECI	TCH_RXI
R/W	R/W	R/W	R/W	R/W	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	TCH_TBEMSK	TCH_TECMSK	TCH_TXMSK
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	TCH_TBEI	TCH_TECI	TCH_TXI
R/W	R/W	R/W	R/W	R/W	R	R	R

Table 23-106 ETHAnCINTMS register contents

Bit position	Bit name	Function
26	TCH_RBEMSK	This bit specifies whether to mask the TCH_RBEI interrupt indicated by bit 18. 0: Does not mask interrupt. 1: Masks interrupt.
25	TCH_RECMSK	This bit specifies whether to mask the TCH_RECI interrupt indicated by bit 17. 0: Does not mask interrupt. 1: Masks interrupt.
24	TCH_RXMSK	This bit specifies whether to mask the TCH_RXI interrupt indicated by bit 16. 0: Does not mask interrupt. 1: Masks interrupt.
18	TCH_RBEI	This bit indicates whether a receive data buffer access error interrupt of INTSCRXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
17	TCH_RECI	This bit indicates whether a receive DMA end of chain interrupt of INTSCRXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
16	TCH_RXI	This bit indicates whether a packet reception DMA transfer completion interrupt of INTSCRXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
10	TCH_TBEMSK	This bit specifies whether to mask the TCH_TBEI interrupt indicated by bit 2. 0: Does not mask interrupt. 1: Masks interrupt.
9	TCH_TECMSK	This bit specifies whether to mask the TCH_TECI interrupt indicated by bit 1. 0: Does not mask interrupt. 1: Masks interrupt.
8	TCH_TXMSK	This bit specifies whether to mask the TCH_TXI interrupt indicated by bit 0. 0: Does not mask interrupt. 1: Masks interrupt.
2	TCH_TBEI	This bit indicates whether a transmit data buffer access error interrupt of INTSCTXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
1	TCH_TECI	This bit indicates whether a transmit DMA end of chain interrupt of INTSCTXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
0	TCH_TXI	This bit indicates whether a packet transmission DMA transfer completion interrupt of INTSCTXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.

(3) ETHAnCTRANSCTL - Transmit checksum transfer control register

This register is used to control transmission/reception of the DMAC for the transmit checksum.

Access This register can be read or written in 32-bit units. Bits 25 and 24 can only be read, however.

Address <ETHAnC_base> + 0308_H

Initial value $0003\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	TCH_RXEN_STA	TCH_TXEN_STA
R/W	R/W	R/W	R/W	R/W	R/W	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	TCH_RXEN	TCH_TXEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-107 ETHAnCTRANSCTL register contents

Bit position	Bit name	Function
25	TCH_RXEN_STA	This bit indicates the reception status. 0: Reception is not in progress (IDLE status). 1: Reception is in progress. Reception stops if a TCH_RBEI or TCH_RECI interrupt, which is controlled by the transmit checksum interrupt register (ETHAnCINTMS), is generated. At this time, this bit is cleared.
24	TCH_TXEN_STA	This bit indicates the transmission status. 0: Transmission is not in progress (IDLE status). 1: Transmission is in progress. Transmission stops if a TCH_TBEI or TCH_TECI interrupt, which is controlled by the transmit checksum interrupt register (ETHAnCINTMS), is generated. At this time, this bit is cleared.
17	TCH_RXEN	This bit specifies whether to enable reception. 0: Disables reception (DMA reception stops.) 1: Enables reception.
16	TCH_TXEN	This bit specifies whether to enable transmission. 0: Disables transmission (DMA transmission stops). 1: Enables transmission.

(4) ETHAnCSFTRST - Transmit checksum software reset register

This register is used to execute a software reset for the DMAC for the transmit checksum.

Access This register can be read or written in 32-bit units.

Address <ETHAnC_base> + $030C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	TCH_ SFTRST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-108 ETHAnCSFTRST register contents

Bit position	Bit name	Function
0	TCH_SFTRST	This is a software reset bit for the DMAC for the transmit checksum. When this bit is set, the DMAC for the transmit checksum and the transmit checksum unit are reset. This bit is automatically cleared after the reset is executed.

(5) ETHAnCDMACM - Transmit checksum DMA control mode setting register

This register is used to specify the burst transfer type of DMA by the DMAC for the transmit checksum.

Access This register can be read or written in 32-bit units.

Address <ETHAnC_base> + 0310_H

Initial value $0000\ 0010_H$. This register is initialized by any reset.

<R>

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	BURST2	BURST1	BURST0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-109 ETHAnCDMACM register contents

Bit position	Bit name	Function
10:8	BURST [2:0]	These bits specify the type of burst transfer for the internal system bus.

BURST2	BURST1	BURST0	Transfer type
0	0	0	Single transfer mode
0	1	1	4-beat incremental burst transfer mode
1	0	0	8-beat incremental burst transfer mode
1	1	1	16-beat incremental burst transfer mode
Other than above			Setting prohibited

(6) ETHAnCRXDP - Transmit checksum receive descriptor pointer register

This register is used to specify the pointer position of the receive descriptor of the DMAC for the transmit checksum. The lower 2 bits are fixed to 00_B.

Access This register can be read or written in 32-bit units.

Address <ETHAnC_base> + 0320_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TCH_RXDP31	TCH_RXDP30	TCH_RXDP29	TCH_RXDP28	TCH_RXDP27	TCH_RXDP26	TCH_RXDP25	TCH_RXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TCH_RXDP23	TCH_RXDP22	TCH_RXDP21	TCH_RXDP20	TCH_RXDP19	TCH_RXDP18	TCH_RXDP17	TCH_RXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TCH_RXDP15	TCH_RXDP14	TCH_RXDP13	TCH_RXDP12	TCH_RXDP11	TCH_RXDP10	TCH_RXDP9	TCH_RXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TCH_RXDP7	TCH_RXDP6	TCH_RXDP5	TCH_RXDP4	TCH_RXDP3	TCH_RXDP2	TCH_RXDP1	TCH_RXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 23-110 ETHAnCRXDP register contents

Bit position	Bit name	Function
31 to 0	TCH_RXDP[31:0]	These bits specify the pointer position of the receive descriptor. Specify the first address of the receive descriptor chain. Bits 1 and 0 are fixed to 00.

(7) ETHAnCLSTRXDP - Transmit checksum last receive descriptor pointer register

This register indicates the last receive descriptor address of the DMAC for the transmit checksum. The lower 2 bits are fixed to 00_B.

Access This register can be read or written in 32-bit units.

Address <ETHAnC_base> + 0324_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TCH_LSTRXDP31	TCH_LSTRXDP30	TCH_LSTRXDP29	TCH_LSTRXDP28	TCH_LSTRXDP27	TCH_LSTRXDP26	TCH_LSTRXDP25	TCH_LSTRXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TCH_LSTRXDP23	TCH_LSTRXDP22	TCH_LSTRXDP21	TCH_LSTRXDP20	TCH_LSTRXDP19	TCH_LSTRXDP18	TCH_LSTRXDP17	TCH_LSTRXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TCH_LSTRXDP15	TCH_LSTRXDP14	TCH_LSTRXDP13	TCH_LSTRXDP12	TCH_LSTRXDP11	TCH_LSTRXDP10	TCH_LSTRXDP9	TCH_LSTRXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TCH_LSTRXDP7	TCH_LSTRXDP6	TCH_LSTRXDP5	TCH_LSTRXDP4	TCH_LSTRXDP3	TCH_LSTRXDP2	TCH_LSTRXDP1	TCH_LSTRXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 23-111 ETHAnCLSTRXDP register contents

Bit position	Bit name	Function
31 to 0	TCH_LSTRXDP[31:0]	These bits indicate the last receive descriptor address. They hold the address of the last accessed receive descriptor. Bits 1 and 0 are fixed to 00.

(8) ETHAnCTXDP - Transmit checksum transmit descriptor pointer register

This register is used to specify the pointer position of the transmit descriptor of the DMAC for the transmit checksum. The lower 2 bits are fixed to 00_B.

Access This register can be read or written in 32-bit units.

Address <ETHAnC_base> + 0328_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TCH_TXDP31	TCH_TXDP30	TCH_TXDP29	TCH_TXDP28	TCH_TXDP27	TCH_TXDP26	TCH_TXDP25	TCH_TXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TCH_TXDP23	TCH_TXDP22	TCH_TXDP21	TCH_TXDP20	TCH_TXDP19	TCH_TXDP18	TCH_TXDP17	TCH_TXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TCH_TXDP15	TCH_TXDP14	TCH_TXDP13	TCH_TXDP12	TCH_TXDP11	TCH_TXDP10	TCH_TXDP9	TCH_TXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TCH_TXDP7	TCH_TXDP6	TCH_TXDP5	TCH_TXDP4	TCH_TXDP3	TCH_TXDP2	TCH_TXDP1	TCH_TXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 23-112 ETHAnCTXDP register contents

Bit position	Bit name	Function
31 to 0	TCH_TXDP[31:0]	These bits specify the pointer position of the transmit descriptor. Specify the first address of the transmit descriptor chain. Bits 1 and 0 are fixed to 00.

(9) ETHAnCLSTTXDP - Transmit checksum last transmit descriptor pointer register

This register indicates the last transmit descriptor address of the DMAC for the transmit checksum. The lower 2 bits are fixed to 00_B.

Access This register can be read or written in 32-bit units.

Address <ETHAn_base> + $032C_H$

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
TCH_LSTTXDP31	TCH_LSTTXDP30	TCH_LSTTXDP29	TCH_LSTTXDP28	TCH_LSTTXDP27	TCH_LSTTXDP26	TCH_LSTTXDP25	TCH_LSTTXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TCH_LSTTXDP23	TCH_LSTTXDP22	TCH_LSTTXDP21	TCH_LSTTXDP20	TCH_LSTTXDP19	TCH_LSTTXDP18	TCH_LSTTXDP17	TCH_LSTTXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TCH_LSTTXDP15	TCH_LSTTXDP14	TCH_LSTTXDP13	TCH_LSTTXDP12	TCH_LSTTXDP11	TCH_LSTTXDP10	TCH_LSTTXDP9	TCH_LSTTXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TCH_LSTTXDP7	TCH_LSTTXDP6	TCH_LSTTXDP5	TCH_LSTTXDP4	TCH_LSTTXDP3	TCH_LSTTXDP2	TCH_LSTTXDP1	TCH_LSTTXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 23-113 ETHAnCLSTTXDP register contents

Bit position	Bit name	Function
31 to 0	TCH_LSTTXDP[31:0]	These bits indicate the last transmit descriptor pointer address. They hold the address of the last accessed transmit descriptor. Bits 1 and 0 are fixed to 00.

23.5 MAC/FIFO/DMAC Function

23.5.1 Frame format

With Ethernet/IEEE802.3, data is transmitted and received as a packet or frame.

The Ethernet controller supports the following three types of frames.

- Basic frame
- VLAN frame
- Pause control frame

(1) Basic frame

The basic frame used with Ethernet consists of a preamble (PA), frame start delimiter (SFD), destination address (DA), source address (SA), type/length field (TYPE/LEN), data field (DATA), and frame check sequence (FCS).

The packet size is defined to be 64 bytes minimum and 1,518 bytes maximum, excluding the preamble (PA) and frame start delimiter (SFD).

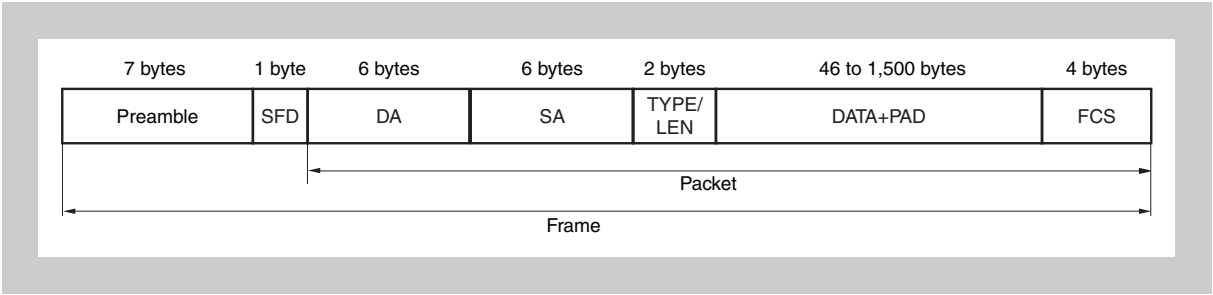


Figure 23-4 Basic frame structure

(a) Preamble and frame start delimiter (SFD)

The preamble and SFD consist of a repetition of 10 for 62 bits followed by 11 at the end, indicating the header of each frame.

(b) Destination address (DA)

The destination address field indicates the destination MAC address. A unicast address, multicast address, or broadcast address is written in this field.

(c) Source address (SA)

The source MAC address is written in this field.

(d) Type/length field

As an Ethernet frame, this field is a type field that indicates a protocol type. As an IEEE802.3 frame, this field is a length field that indicates the length of the data field.

(e) Data field

The data field size ranges from 46 bytes to 1,500 bytes.

Depending on the communication protocol, the data field might be divided to insert special header information. The Ethernet controller uses the data in this field only for calculating the CRC (Cycle Redundancy Check) for the FCS and does not check its contents.

(f) Frame check sequence (FCS)

The frame check sequence field is used to write a 32-bit CRC code to check the transfer data.

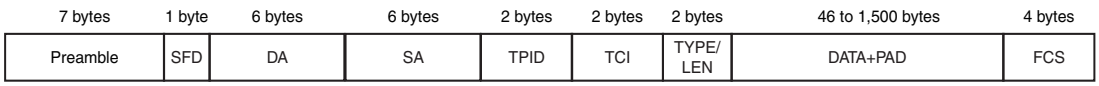
The Ethernet controller can automatically append a CRC to a transmit frame.

(2) VLAN frame

The structure of a VLAN frame (Qtag frame) is slightly different from that of a basic frame.

A 4-byte VLAN header is inserted immediately after the source address field. As a result, the minimum packet length of a VLAN frame is 64 bytes and the maximum packet length is 1,522 bytes.

The Ethernet controller has a VLAN frame detection function. If a transmit or receive packet is detected to be a VLAN frame, packet processing is performed based on the receive packet length.



Note TPID: Tag Protocol ID
TCI: Tag Control Information
TPID + TCI: VLAN header

Figure 23-5 VLAN frame structure

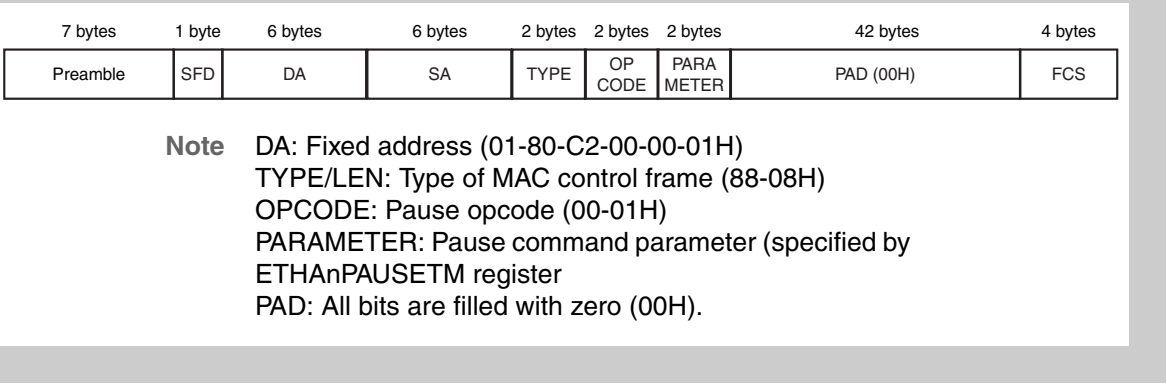
Caution The Ethernet controller recognizes the value set to the ETHAnVLTP.VLTP[15:0] bits as a VLAN frame (TPID). The default value is 0000_H. For details, see *(10) "ETHAnVLTP - VLAN type register" on page 1626 in 23.4.1 "MAC control registers"*.

(3) Pause control frame

The pause control frame is a 64-byte packet that has a dedicated format.

The destination address field has a fixed value of 01-80-C2-00-00-01_H.

The type/length field has a value of 8808H, which indicates a control frame, and the opcode has a value of 0001H, which indicates a pause control frame. The parameter field stores the value specified by the ETHAnPAUSETM register. The unused area following the parameter field is filled with PAD data consisting of zeros.



Note DA: Fixed address (01-80-C2-00-00-01H)
TYPE/LEN: Type of MAC control frame (88-08H)
OPCODE: Pause opcode (00-01H)
PARAMETER: Pause command parameter (specified by ETHAnPAUSETM register
PAD: All bits are filled with zero (00H).

Figure 23-6 Pause control frame structure

The Ethernet controller can automatically transmit a pause control frame according to the amount of the remaining in the receive FIFO.

When a frame is received, the frame type is identified by the DA, TYPE, and OPCODE fields as shown in *Table 23-114 "Frame reception"*.

Table 23-114 Frame reception

DA	TYPE	OPCODE	Frame identified
01-80-C2-00-00-01	8808_H	0001_H	Pause frame
01-80-C2-00-00-01	8808_H	Other than 0001_H	Not supported
01-80-C2-00-00-01	Other than 8808_H	xxxx	Data frame
Unicast (station address)	8808_H	0001_H	Pause frame
Unicast (station address)	8808_H	Other than 0001_H	Not supported
Unicast (station address)	Other than 8808_H	xxxx	Data frame
Multicast	8808_H	xxxx	Not supported
Multicast	Other than 8808_H	xxxx	Data frame
Unicast (station address)	8808_H	xxxx	Not supported
Unicast (station address)	Other than 8808_H	xxxx	Data frame

(4) Pause control frame containing VLAN tag

The Ethernet controller does not support a pause control frame containing a VLAN tag.

It receives such a frame as an ordinary VLAN packet, but the RBRO and RLOR flags of the reception status monitor register (ETHAnRXSTMONI) are set (the RPCF and RCFR flags are not set).

(5) Envelope frame

The envelope frame format was added to IEEE802.3as (2005). Because it is a frame for 1,000 Mbps half-duplex communication, the Ethernet controller does not support this type of frame. If an envelope frame containing an EXTENSION field is received, a CRC error or length field mismatch occurs, or the receive FIFO overflows. Check the reception status and discard such a frame.

The Ethernet controller cannot transmit an envelope frame containing an EXTENSION field.

23.5.2 Frame transmission

The Ethernet controller generates a transmit frame as defined in IEEE802.3 from the transmit packet data stored in the transmit FIFO by the DMAC for the Ethernet controller via a DMA transfer, and then outputs that frame to a PHY device. If a collision is detected, the frame is retransmitted by using a random back-off algorithm. The status information of each transmit frame, such as excessive transmit delays and collisions exceeding the maximum number, is reflected in the ETHAnTXSTATUS register, and the number of times each event has occurred in all transmit frames is counted by statistics counters.

(1) Transmit frame

The transmit frame defined by IEEE802.3 consists of the following six fields (see *Figure 23-4 "Basic frame structure"*).

- Preamble (PA)
- Frame start delimiter (SFD)
- Destination address (DA)
- Source address (SA)
- Length field (LEN)
- Data and frame check sequence (FCS)

During transmission, the Ethernet controller generates a preamble, frame start delimiter, and FCS data.

(2) Transmission clock

The Ethernet controller operates in synchronization with the transmission clock (TXCLK) supplied by an external PHY device. Transmit packet data stored in the transmit FIFO via a DMA transfer is synchronized with TXCLK in the FIFO buffer, and then output to the PHY device. IEEE802.3 defines the frequency of TXCLK as 25 MHz±100 ppm at a data transfer rate of 100 Mbps, or 2.5 MHz±100 ppm at 10 Mbps.

(3) Carrier sense signal (CRS)

During half-duplex communication, if a carrier is detected (CRS = 1) after the Ethernet controller has stored transmit data in the FIFO buffer and transmission is enabled, the Ethernet controller postpones transmission until the end of the carrier (CRS = 0). After the carrier ends, transmission starts when the inter-packet gap (IPG) count specified by the ETHAnIPGT register has been reached.

If no carrier is detected (CRS = 0) when transmission is enabled and the IPG count is reached after the end of the previous carrier, transmission starts immediately.

When a frame is transmitted from the local side, the transmitted carrier sense signal is looped back from the external PHY device. If the transmitted carrier sense signal is masked by the external PHY device in the user-defined system,

the Ethernet controller detects a carrier sense error, but this does not affect the transmission.

(4) Collision detection (COL) and retransmission

If the Ethernet controller detects a collision during half-duplex communication, it transmits jam data (an error CRC) and then stops transmission.

If less than the maximum number of collisions (default value: 15) are detected in the collision window, transmission is kept waiting by a random back-off algorithm and data in the transmit FIFO is retransmitted. (In this case, no data is captured into the FIFO buffer again by DMA.)

If collisions exceeding the maximum number are detected or if a late collision (a collision detected outside the collision window) occurs, transmission is aborted and the transmit data is discarded.

(5) Inter-packet gap (IPG)

The IPG for successive data transmission from the local side is specified by the ETHAnIPGT register, and that for other cases is specified by the ETHAnIPGR register.

When transmission from the local side or the communicating device is finished, the Ethernet controller starts counting the IPG. If a request for the next transmission is issued from the FIFO buffer after transmission from the local side is finished and before the IPG count reaches the value of the ETHAnIPGT register, it is assumed that the data is to be transmitted successively (back to back), and transmission starts as soon as counting finishes.

If a packet is to be transmitted after transmission from the communicating device is finished, the IPG count is controlled by the ETHAnIPGR register. In the ETHAnIPGR register, the entire period of the IPG is specified for the IPGR2 bits and the interval time to detect a carrier in the first half of the IPG is specified for the IPGR1 bits. If a carrier is detected during the period specified for the IPGR1 bits, the Ethernet controller waits for the end of the carrier and then starts IPG counting from the beginning. If no carrier is detected during the period specified for the IPGR1 bits, transmission starts after the IPG period specified for the IPGR2 bits elapses.

If transmission does not start within the time required to transmit 24,288 bits (2.43 ms at 10 Mbps or 243.88 μs at 100 Mbps) after the next transmission request is received from the FIFO buffer, an excessive transmission delay is assumed, transmission is aborted, and the transmit data is discarded.

The set value of the ETHAnIPGT and ETHAnIPGR registers and the actual IPG period are calculated by the following expression.

[At 100 Mbps]

Back-to-back transmission:

IPG = (5 + IPGT) × 40 ns (default value: 960 ns)

Non back-to-back transmission:

IPG = (5 + IPGT) × 40 ns (default value: 960 ns)

Carrier sense time: (2 + IPGR1) × 40 ns (default value: 640 ns)

[At 10 Mbps]

Back-to-back transmission:

IPG = (5 + IPGT) × 400 ns (default value: 9.6 µs)

Non back-to-back transmission:

IPG = (5 + IPGR2) × 400 ns (default value: 9.6 µs)

Carrier sense time: (2 + IPGR1) × 400 ns (default value: 6.4 µs)

Caution According to the specification of IEEE802.3, set the IPG to 960 ns or more at a data transfer rate of 100 Mbps, or 9.6 µs or more at 10 Mbps.

The default value of the ETHAnIPGT and ETHAnIPGR registers is the minimum rated value and may be used as is.

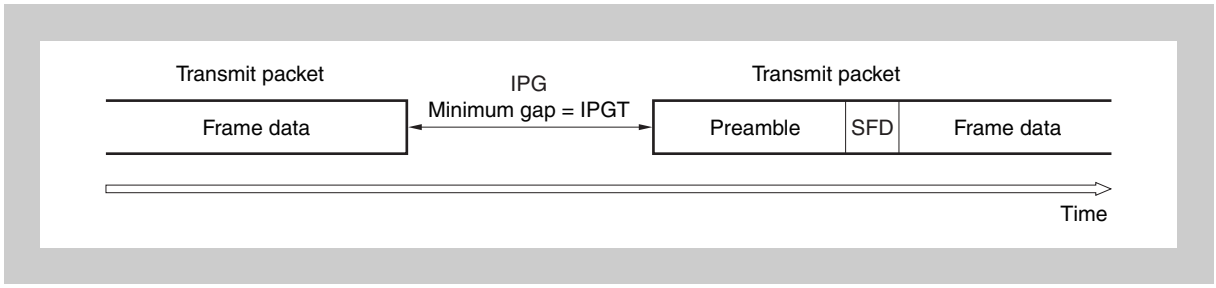


Figure 23-7 IPG during back-to-back transmission

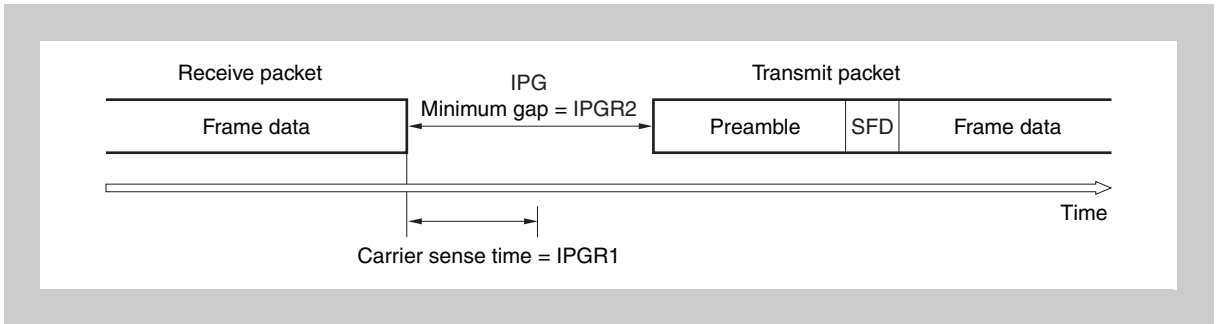


Figure 23-8 IPG during non back-to-back transmission

(6) Appending preamble, CRC, and PAD

A 7-byte preamble and a 1-byte frame start delimiter (SFD) are appended to the beginning of the transmit packet supplied from the FIFO buffer.

ETHAnMACC1.CRCEN	Operation
0	The transmit packet must end with a valid frame check sequence (FCS). The MAC checks the FCS, and if the FCS value is not correct, the MAC generates a transmission status interrupt (INTETMTS) to report an error.
1	An internally generated frame check sequence (FCS) is appended to the end of the transmit packet.

If the ETHAnMACC1.CRCEN bit is set, a frame check sequence (FCS) that has been internally generated is appended to the end of the transmit packet.

If the ETHAnMACC1.CRCEN bit is cleared, the transmit packet must end with a valid FCS. The Ethernet controller can check the FCS. If the value of the FCS is not correct, the Ethernet controller generates an Ethernet transmission status interrupt.

If the ETHAnMACC1.PADEN bit is set, zeros (PAD) are appended to a transmit packet shorter than 64 bytes (this is known as padding). In this case, the Ethernet controller appends the correct FCS to the end of the frame, regardless of the setting of the CRCEN bit.

If the ETHAnMACC2.APD or ETHAnMACC2.VPD bit is set when the ETHAnMACC1.PADEN bit is 1, PAD is appended to a VLAN frame. If the APD bit is set, only a packet that matches the VLAN type specified by the ETHAnVLTP register is regarded as a VLAN frame and is padded. If the VPD bit is set, all frames are regarded as VLAN frames and padded. A packet regarded as a VLAN frame is padded to extend its length to 68 bytes. The data that is appended as a pad is all 0.

(7) Aborting transmission

The Ethernet controller aborts transmission under the following conditions.

It does not abort transmission if the transmit FIFO underruns within the normal operating range.

- If more than the maximum number of collisions (MAX collision) occur
- If collision occurs outside the collision window (late collision)
- If there is an excessive transmission delay
- If a packet exceeding the frame length specified by the ETHAnLMAX register is to be transmitted.

  (If the ETHAnMACC1.HUGEN bit is 1, however, the transmit frame length is not limited.)

(8) Full-duplex operation

A full-duplex operation is enabled when the ETHAnMACC1.FULLD bit is set. The IPG is always the value specified by the ETHAnIPGT register. The FULLD signal is asserted if the ETHAnMACC1.FULLD bit is set, reporting to the external circuit that full-duplex operation is specified.

(9) register details functions

The Ethernet controller has a flow control function (see *(1) "Flow control" on page 1746 in 23.5.4 "MAC control function"*) and a back pressure function (see *(2) "Back pressure" on page 1748 in 23.5.4 "MAC control function"*) associated with the receive FIFO. These functions are automatically activated to prevent the FIFO buffer from overflowing when the vacant capacity of the receive FIFO runs short.

(10) Transmission status update timing

The transmission status is updated in the timing shown below.

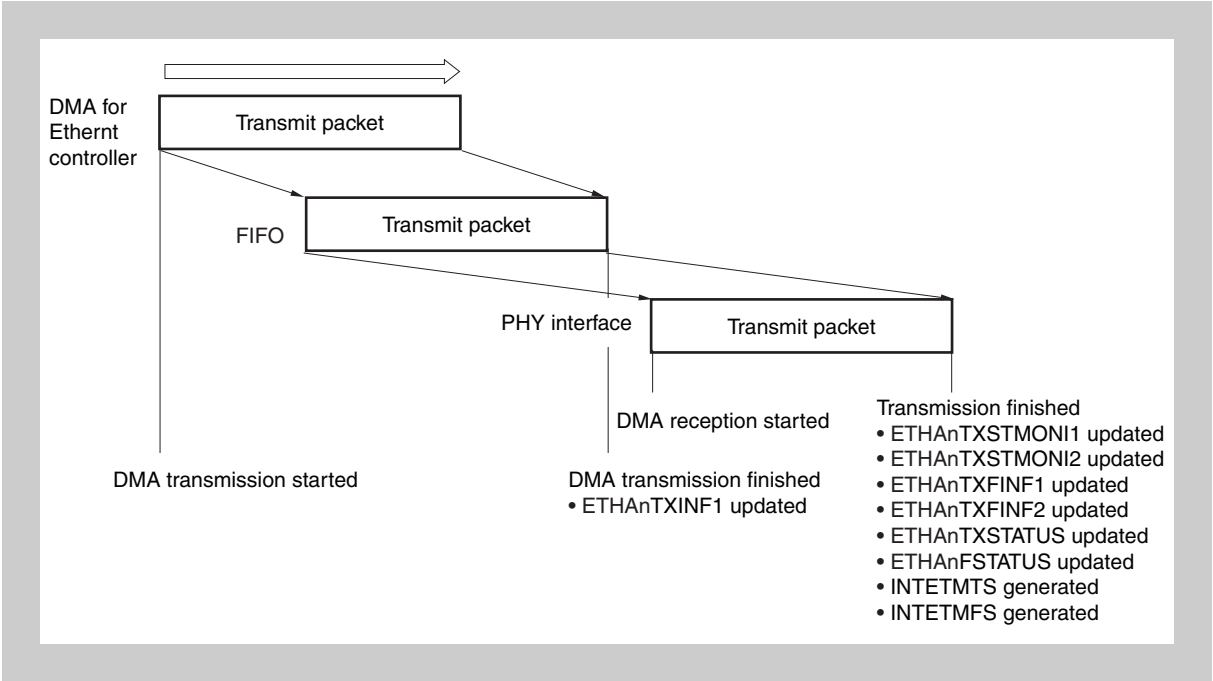


Figure 23-9 Transmission status update timing

23.5.3 Frame reception

The Ethernet controller generates a receive packet from a receive frame to be stored in the FIFO buffer, detects the SFD, checks the length field and FCS, and identifies whether the frame is a VLAN frame.

The status information of each receive packet is set to the reception status monitor register (ETHAnRXSTMONI) and the number of times each event has occurred in all receive frames is counted by statistics counters.

(1) Receive clock

The Ethernet controller receives data in synchronization with the reception clock (RXCLK) supplied by an external (PHY) device.

IEEE802.3 defines the frequency of RXCLK as 25 MHz±100 ppm at a data transfer rate of 100 Mbps, or 2.5 MHz±100 ppm at 10 Mbps.

(2) Reception of MII data

The Ethernet controller recognizes the RXD signal data as receive frames while the RXD[3:0] signal is asserted, and recognizes the end of the frames when the RXDV signal is deasserted.

(3) Detecting preamble and SFD

The Ethernet controller detects the preamble and SFD at the beginning of a receive frame and recognizes the data that follows as a receive packet.

(4) Checking length field

The Ethernet controller counts the length of a receive packet, and checks the length of the data field assuming the 2 bytes following the source address to be a length field. The result of this check can be read as the reception status from the ETHAnRXSTMONI register. An interrupt signal can be output to report a mismatch between the counted receive packet length and the length field value.

(5) CRC

The Ethernet controller calculates the 4-byte frame check sequence (FCS) from a receive packet and compares it with the FCS data appended to the end of the receive packet. The result of the comparison can be read from the ETHAnRXSTMONI register. An interrupt signal can be output to report a mismatch between the calculated 4-byte FCS and the FCS data at the end of the receive packet.

(6) Transmitting data to FIFO

The Ethernet controller assumes that a packet of 6 bytes or more is valid and discards a packet of less than 6 bytes.

(7) Detection of huge packet

If the ETHAnMACC1.HUGEN bit is cleared, the Ethernet controller receives only packets shorter than the maximum frame length specified by the ETHAnLMAX register (default value: 1,536 bytes), and stops receiving longer packets midway.

For details about the receivable packet length, see *Table 23-122 "Restrictions on receive FIFO"*.

(8) Detecting VLAN frame

The Ethernet controller checks whether received packets are VLAN frames.

If the value of the TPID field (the 2 bytes following the source address) of a received packet matches the value specified by the ETHAnVLTP register, the packet is recognized as a VLAN packet and the ETHAnRXSTMONI.VLAN flag is set. In a packet recognized as a VLAN frame, the 2 bytes immediately after the VLAN header (the 4 bytes following the source address) including the TPID field are regarded as the length field.

(9) Reception status update timing

The reception status is updated in the timing shown below.

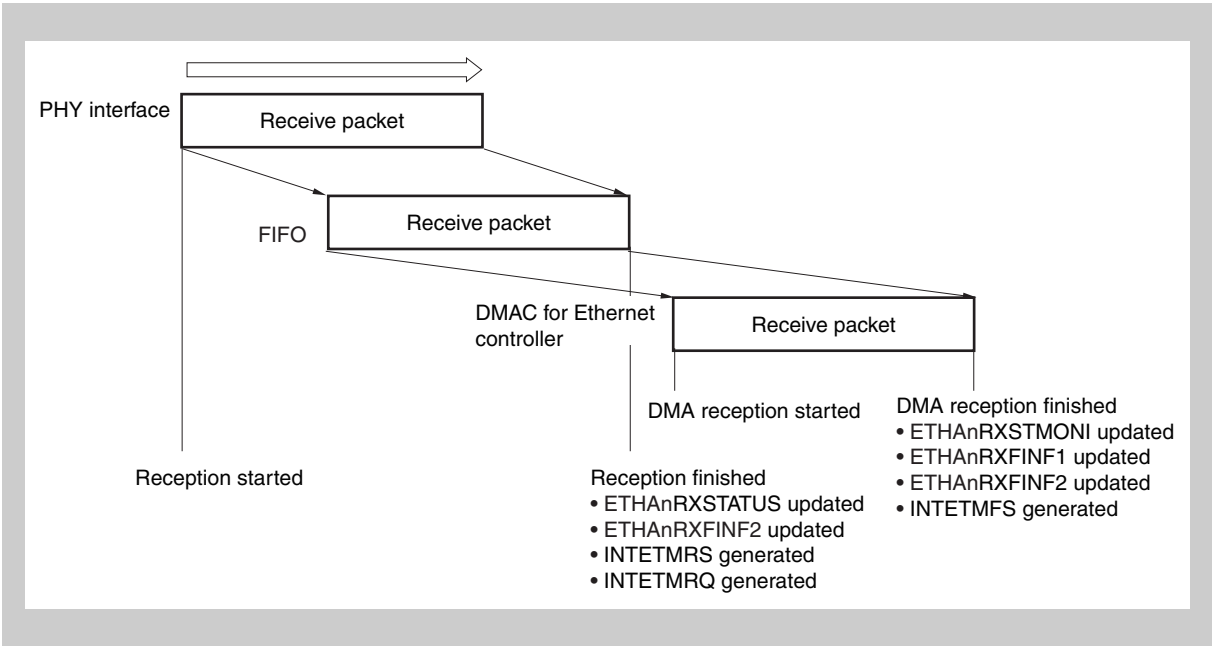


Figure 23-10 Reception status update timing

23.5.4 MAC control function

(1) Flow control

The Ethernet controller controls the flow by using the pause control frame defined in “IEEE802.3 Annex 31”.

The purpose of flow control is to decrease the frequency of frame transmission executed by the communicating device (link partner) connected point-to-point during full-duplex operation. The amount of data a system can receive and process is limited. If frames are received too frequently, system processing cannot keep up, so the receive FIFO might overflow. Flow control is used to avoid this situation.

When the Ethernet controller receives a pause control frame, it loads the value of the parameter field in the control frame to the pause timer in the MAC. If the value of the pause timer is not 0, the next transmission starts after the time set to the pause timer has elapsed.

If the value of the parameter field in the received pause control frame is 0 (a zero pause control frame), the value of the pause timer is cleared to 0 and transmission is resumed after the packet interval specified by the ETHAnIPGR register has elapsed.

To suppress data transmission from the link partner, the reserved multicast address (01-80-C2-00-00-01), pause opcode (00-01), and the pause timer value of the ETHAnPAUSETM register (PAUSETM_MAX) are transmitted as a pause control frame.

Starting transmission of a pause frame takes precedence over starting transmission of a basic frame. If a condition for transmitting a pause frame is satisfied while a basic frame is being transmitted, however, the pause frame is transmitted after transmission of the basic frame has finished.

The Ethernet controller controls the flow by setting ETHAnMFFCONT.FLOWCNT.

The necessity of transmitting a pause control frame request is judged according to the amount of data in the receive FIFO.

If the ETHAnMFFCONT.IVPAUSE bit is 0 in the full-duplex communication mode, the Ethernet controller monitors the amount of data in the receive FIFO during reception (see *(a)* in *Figure 23-11 “Flow control”*). And if the amount of data in the receive FIFO exceeds the value specified by the ETHA0FLOWTH.FLOWTHR bits, a pause control frame is transmitted (see *(b)* in *Figure 23-11 “Flow control”*).

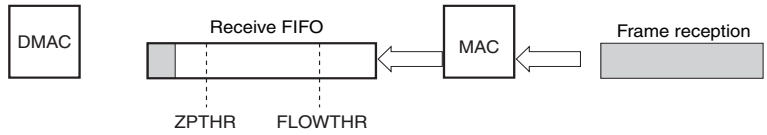
If the ETHAnMFFCONT.IVPAUSE bit is 1, the pause control frame is continually transmitted at the interval specified by the ETHAnPAUSETM.IPTIME bits as long as the amount of data in the receive FIFO exceeds the value specified by the ETHA0FLOWTH.FLOWTHR bits.

The Ethernet controller monitors the quantity of data in the receive FIFO even while receive data is being transferred by DMA (see *(c)* in *Figure 23-11 “Flow control”*).

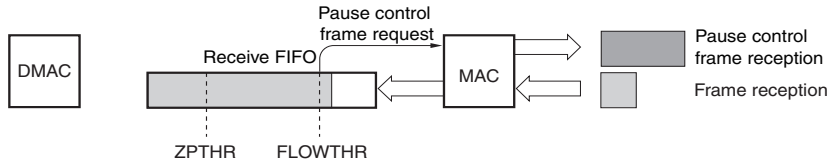
If the ETHAnMFFCONT.ZEROPAUSE bit is 1, a zero pause control frame is transmitted if the amount of data in the receive FIFO falls below the value specified by the ETHA0FLOWTH.ZPTHR bits (see *(d)* in *Figure 23-11 “Flow control”*).

If the ETHAnMFFCONT.ZEROPAUSE bit is 0, a zero pause control frame is not transmitted.

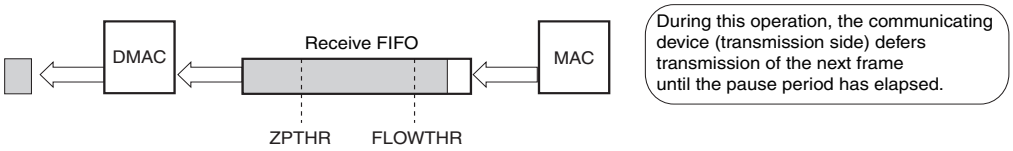
(a) When a frame is received, data is stored in the receive FIFO.



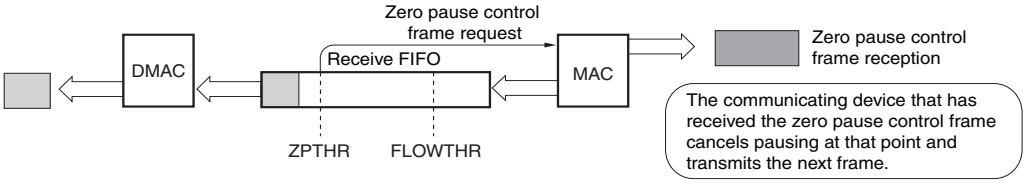
(b) When the amount of data in the receive FIFO exceeds the value specified by FLOWTHR, a pause control frame is transmitted.



(c) The data in the receive FIFO is read by DMA.



(d) When the amount of data in the receive FIFO falls below the value specified by ZPTHR, a zero pause control frame is transmitted.



Note The shaded part ▭ of the receive FIFO in the above figures indicates the amount of data in the receive FIFO, not how data is actually stored.

Figure 23-11 Flow control

(2) Back pressure

This function is available only during half-duplex operation.

If the amount of data in the receive FIFO exceeds the value specified by the ETHA0FLOWTH.FLOWTHR bits when the ETHAnMFFCONT.FLOWCNT bit is 1 and the ETHAnMACC1.FULLD bit is 0, the back pressure function is enabled (see *(b)* in *Figure 23-12 "Back pressure control"*).

If the next frame is received in this status, a collision is intentionally generated by transmitting a dummy packet, prompting the communicating device to retransmit the frame (see *(c)* in *Figure 23-12 "Back pressure control"*).

A collision that occurs in the back pressure status is not included in the number of collisions.

The back pressure status is released if the amount of data in the receive FIFO falls below the value specified by the FLOWTHR bits (see *(d)* in *Figure 23-12 "Back pressure control"*).

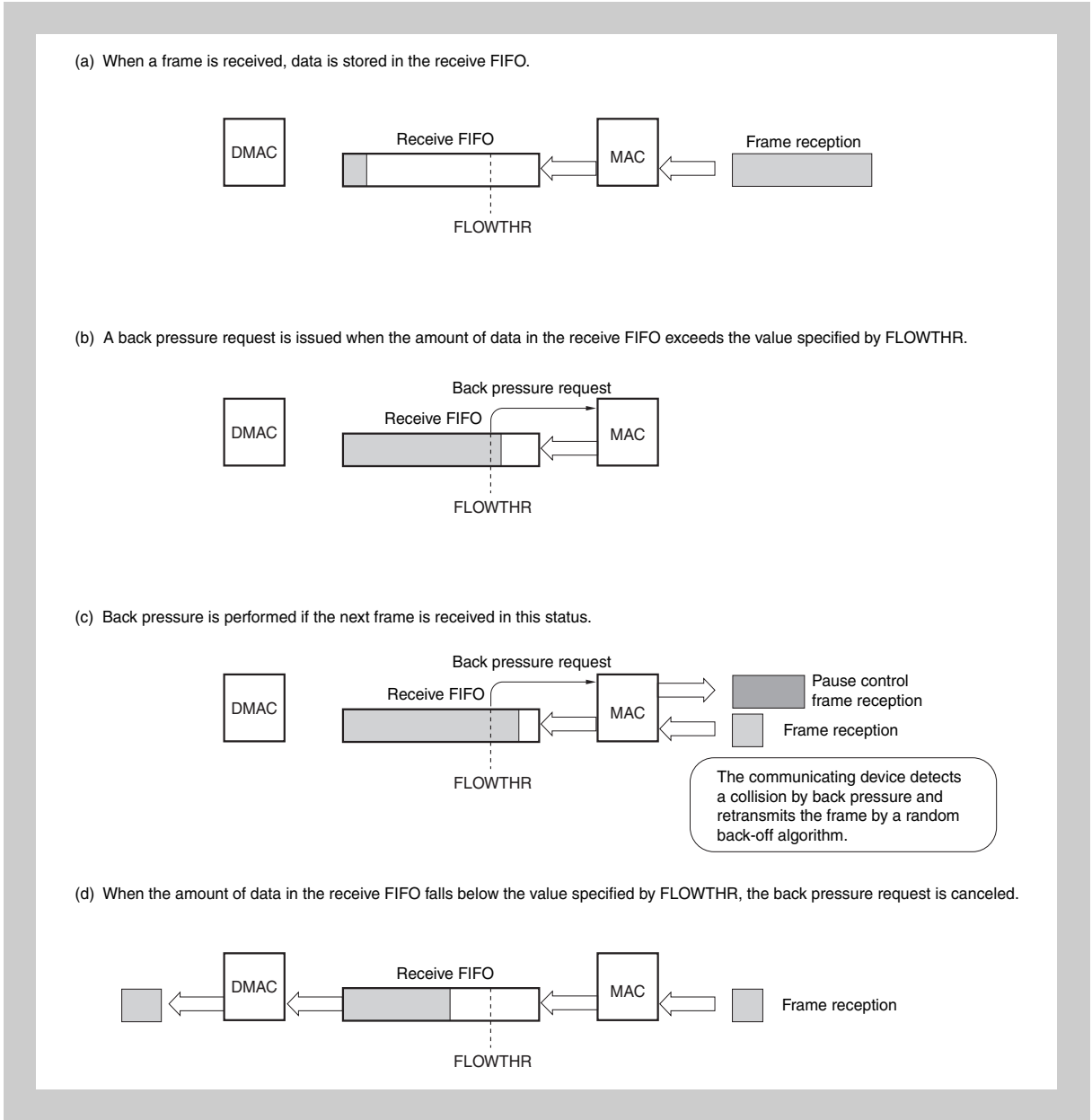


Figure 23-12 Back pressure control

(3) Operations related to VLAN frame

The Ethernet controller detects a VLAN frame by comparing the TPID field in a receive/transmit packet with the value of the VLAN type register (ETHAnVLTP). Operations related to the VLAN frame are described below.

(a) Detection of VLAN frame

The Ethernet controller constantly monitors the value of the 2-byte TPID field that follows the source address in a receive packet.

During transmission, the value of the TPID field is checked when the ETHAnMACC2.APD or ETHAnMACC2.VPD bit is 1.

The Ethernet controller recognizes a packet whose TPID field matches the value of the VLAN type register (ETHAnVLTP) as a VLAN frame.

(b) Reception of VLAN frame

If the value of the TPID field of a received packet matches the value of the VLAN type register (ETHAnVLTP), all judgments concerning the frame size are made based on the VLAN frame size (MAX: 1,522 bytes, MIN: 64 bytes).

(c) Transmission of VLAN frame

If a frame whose TPID field value matches the value of the VLAN type register (ETHAnVLTP) is transmitted from an upper layer when the ETHAnMACC2.APD bit is 1, the frame is recognized as a VLAN frame and is padded to extend its length to 68 bytes.

When the ETHAnMACC2.VPD bit is 1, all frames are recognized as VLAN frames and are padded with “0” to extend their lengths to 68 bytes.

23.5.5 DMAC

The dedicated DMAC for the Ethernet controller is a DMA function for the internal system bus of the Ethernet controller.

It is dedicated for the transmission/reception Ethernet controller.

All transmission and reception data is transferred by the dedicated DMAC for the Ethernet controller.

(1) DMA transfer mode

The following settings can be specified by using the ETHAnDMACM register.

Transfer mode

- Single transfer mode
- 4-beat incremental burst transfer mode
- 8-beat incremental burst transfer mode
- 16-beat incremental burst transfer mode

After the transfer mode has been specified by the ETHAnDMACM register, it will be applied starting with the next DMA transfer.

Cautions

1. Undefined length burst transfer mode cannot be specified using the register.
   Undefined length burst transfer is automatically used by the Ethernet controller to internally process fractional data during a DMA transfer.
   It is therefore not possible to transfer all transfer data in this mode intentionally.
2. The register that specifies the transfer mode is not locked during a DMA transfer. If the setting of this register is changed during a DMA transfer, therefore, the current DMA cycle becomes illegal. Do not change the setting of the register during a DMA transfer (when RXEN_STA or TXEN_STA is 1).

(2) Areas accessible by DMA transfer

The following area is subject to DMA transfers:

- HBUS-RAM

(3) DMA address boundary

With the DMA for the Ethernet controller, the address boundary does not have to be considered when setting the start address of the data buffer and the number of transfer bytes.

If there is fractional data during a burst transfer, the fraction is automatically processed before the data is transferred.

For reception, however, because it is not possible to predict where the data to be received will end, the last transfer might be a dummy transfer when a burst transfer is used.

Note In the 4-, 8-, or 16-beat incremental burst transfer mode, if the last data is shorter than the fixed length, it is automatically transferred in undefined length burst transfer mode.

Byte access for byte alignment is always executed in the single transfer mode.

(4) DMA arbitration

Because the Ethernet controller supports full-duplex transfer, DMA transmission and DMA reception might be executed together. If DMA requests for transmission and reception are issued at the same time, the reception request takes precedence.

23.5.6 Serial management interface

The Ethernet controller has a pair of serial management interfaces which can be used to set a PHY device, to read statuses, and for communicating with the PHY device when auto-negotiation is used.

Set the address of the PHY device to be connected by Ethernet controller to the ETHAnMADR register before using the serial management interface.

(1) Overview of serial management interface

(a) MDC clock

The management data clock (MDC) is generated by dividing the Ethernet control clock (f_{EC}).

The division ratio is specified by the ETHAnMIIC.CLKS bits.

Table 23-115 ETHAnMIIC register: CLKS bits and f_{EC} frequency

ETHAnMIIC.CLKS bits			Frequency range of f_{EC} input
Bit 4	Bit 3	Bit 2	
0	0	0	Setting prohibited
0	0	1	33 MHz or less
0	1	0	50 MHz or less
0	1	1	66 MHz or less
1	0	0	100 MHz or less
1	0	1	Setting prohibited
1	1	0	Setting prohibited
1	1	1	Setting prohibited

If the ETHAnMIIC.PHYSEL bit is 0 (the default value), the MDC is output only when a management frame is transmitted or received.

The MDC is always output when the ETHAnMIIC.PHYSEL bit is set.

If communication with the PHY device fails when the ETHAnMIIC.PHYSEL bit is 0, set this bit.

(b) Serial management frame structure

The Ethernet controller generates a serial management frame shown below by writing a value to the ETHAnMCMD or ETHAnMWTD register.

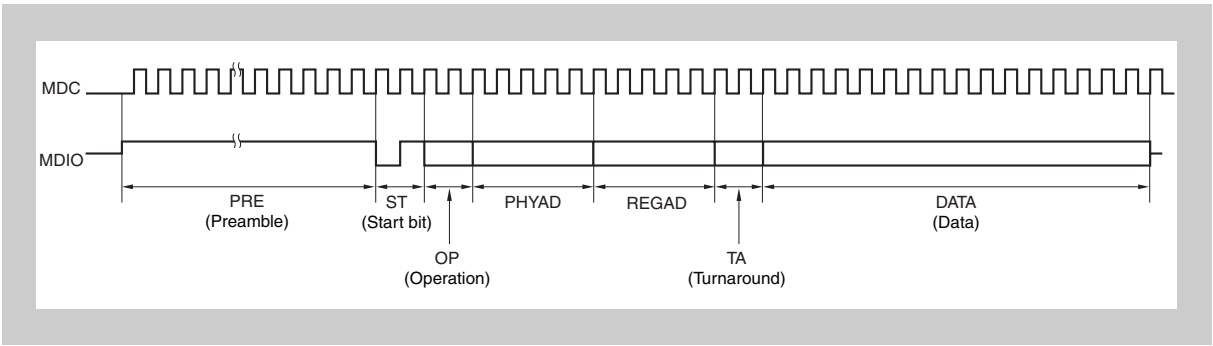
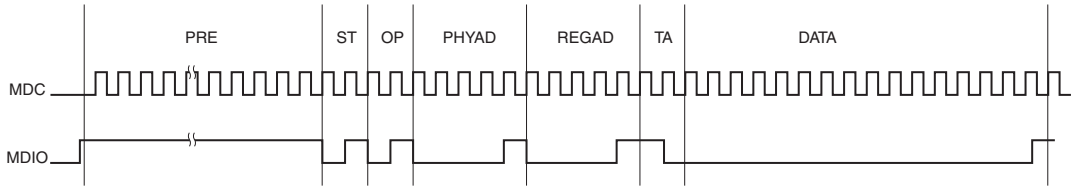


Figure 23-13 Serial management frame structure

A 32-bit preamble, 2-bit start bit field, and 2-bit opcode that indicates whether a register in the PHY device is read or written, are automatically appended to the serial management frame. PHYAD and REGAD indicate the address of the externally connected PHY device and the address of a register in that PHY device, respectively. The values set to the ETHAnMADR.FIAD and ETHAnMADR.RGAD bits are appended to PHYAD and REGAD, respectively.

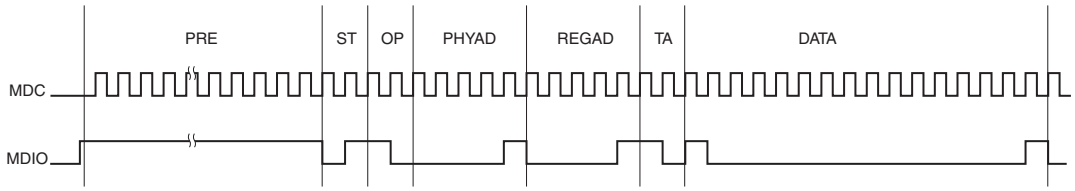
The Ethernet controller serially outputs data from the preamble to REGAD, and after a 2-bit turnaround, the data set to the ETHAnMWTD.CTLD bits is output for a write access. For a read access, serial data is input by the MDI signal and written to the ETHAnMRDD.PRSD bits.

While the MDO signal is being output, the MDOEN signal is asserted to 1.



Note Setting conditions: PHYAD = 01H, REGAD = 01H, CTLD = 0001H

Figure 23-14 Timing of MII management interface signals (write access)



Note Setting conditions: PHYAD = 01H, REGAD = 01H, CTLD = 8001H

Figure 23-15 Timing of MII management interface signals (read access)

(c) SCAN command

The Ethernet controller has a SCAN command to successively read a specific PHY register. When the ETHAnMCMD.SCANC bit is set, read accesses are generated successively. By reading the ETHAnMRDD.PRSD bit, the specific PHY register can be polled.

(2) Procedure for transmitting or receiving a serial management frame

Serial management frames are transmitted and received as follows.

First, the ETHAnMIND.SCANA bit is checked to see whether the SCAN command is under execution.

If not, the ETHAnMIND.BUSY bit is checked to see whether the serial management frame is being accessed. If the BUSY bit is 1, the Ethernet controller waits until it is cleared. If the SCAN command is being executed, the ETHAnMCMD.SCANC bit is cleared and then the Ethernet controller waits until the BUSY bit is cleared.

Next, the address of the external PHY device to which the frame is to be transmitted and the address of a register in the PHY device are set to the ETHAnMADR.FIAD and ETHAnMADR.RGAD bits, respectively.

Write access is started by writing to the ETHAnMWTD.CTLD bits.

The BUSY bit is set when data has been written to the ETHAnMWTD register and cleared when writing has finished.

Read access is started by writing 1 to the ETHAnMCMD.RSTAT bit. When the RSTAT bit is set, the BUSY bit is set. The BUSY bit is cleared after completion of reading. The host system can obtain the data of the PHY register by confirming that the BUSY bit is 0 and then reading the ETHAnMRDD.PRSD bits.

To execute the SCAN command, set the ETHAnMCMD.SCANC bit. While this bit is 1, reading is repeatedly executed. The ETHAnMIND.SCANA bit holds 1 while the SCAN command is executed. The ETHAnMIND.NVALID bit holds 1 until the first read access finishes after the SCAN command is executed. The ETHAnMIND.BUSY bit is set when the SCAN command is executed. If the SCAN command is disabled (by clearing the ETHAnMCMD.SCANC bit), the ETHAnMIND.BUSY bit is cleared after the current read access finishes.

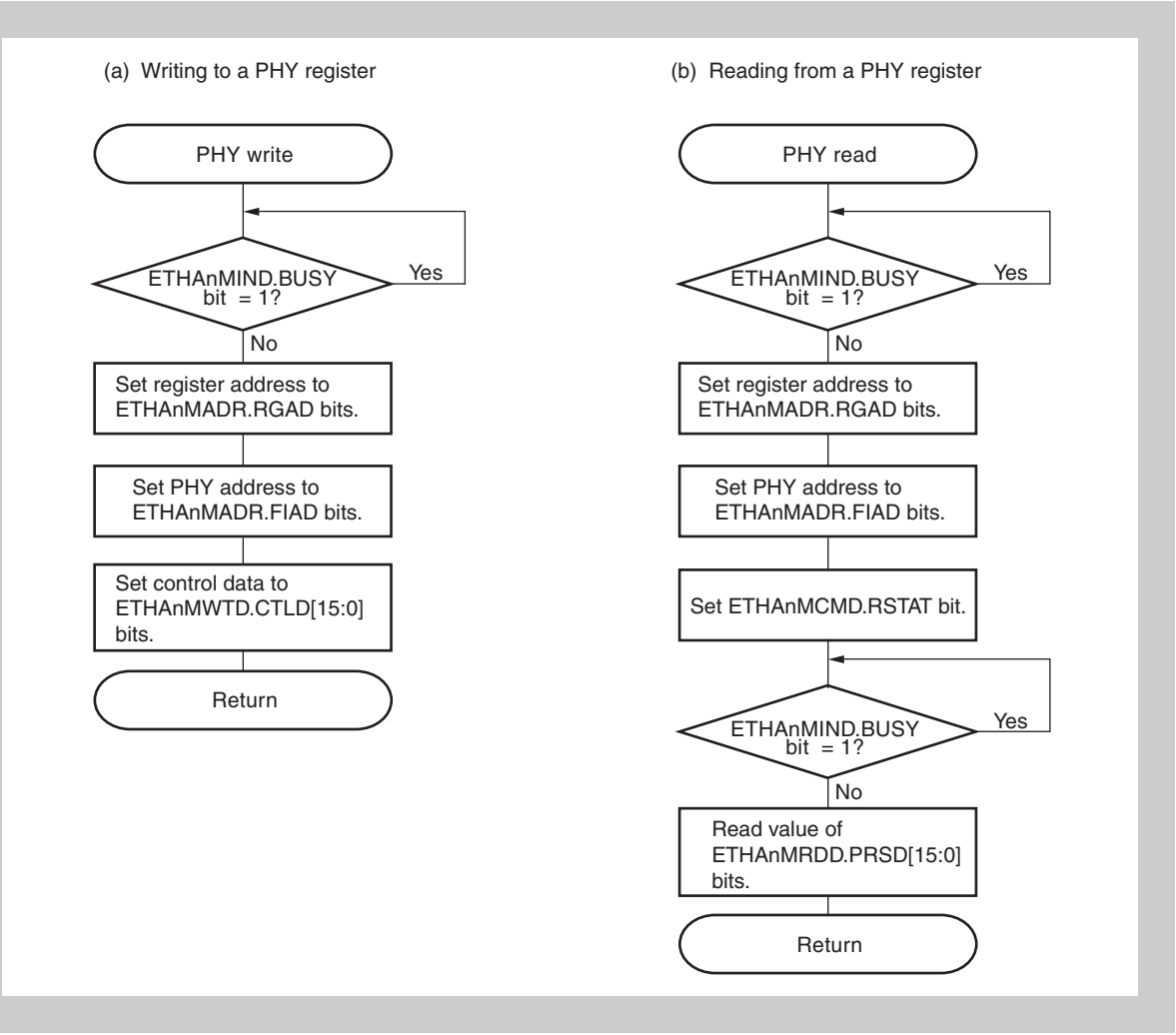


Figure 23-16 Accessing PHY register

23.5.7 Address filtering

(1) Overview of address filtering

The Ethernet controller filters addresses by using the destination address of a received packet and, based on the filtering result, decides whether to accept or discard the received packet.

The filtering conditions can be specified by the ETHAnAFR register. Conditions can be individually specified for unicast addresses, multicast addresses, and broadcast addresses, or conditions can be combined.

(a) Filtering of unicast addresses

The address set to the ETHAnLSA1 and ETHAnLSA2 registers is compared with the destination address of a received packet as a unicast address. A received packet is accepted if its destination address matches the address specified for these registers, and is discarded if not. Each receive packet is checked to see if its destination address matches the set unicast address.

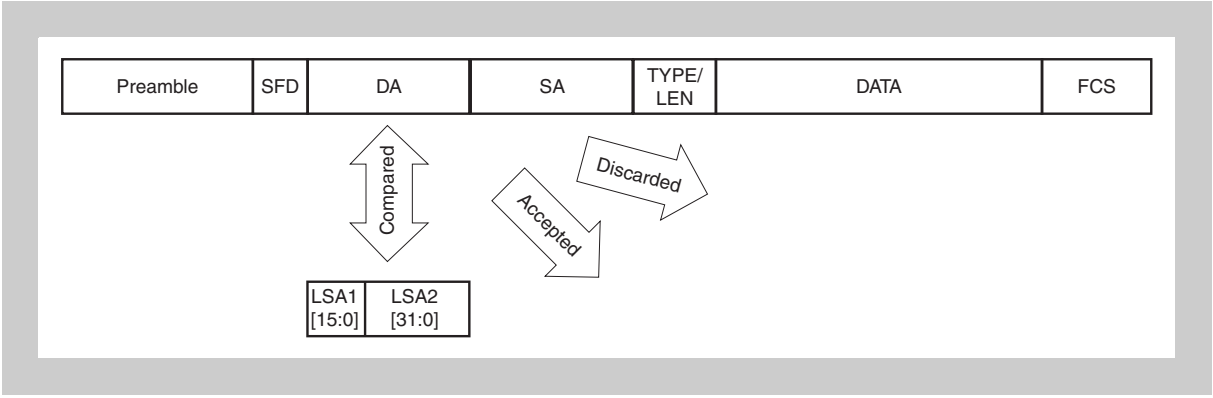


Figure 23-17 Image of filtering by unicast address during reception

(b) Filtering of multicast addresses

A multicast address is filtered in two ways. If the ETHAnAFR.PRM bit is set, all received packets that have a multicast address as the DA are accepted.

If the ETHAnAFR.AMC bit is set, only a received packet that has a multicast address that matches the hash table set up by the ETHAnHT1 and ETHAnHT2 registers is accepted, and a received packet whose multicast address does not match is discarded.

The hash table is used as follows for multicast address filtering.

The hash table is referenced by using bits [28:32] of the 32 bits of the CRC calculation result of the received multicast address. The following polynomial is used for calculating the CRC.

$$CRC(x) = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1$$

If 1 is specified at the bit position indicated by the value resulting from decoding the above 6 bits in the ETHAnHT1 and ETHAnHT2 registers (shown in *Table 23-116 "Correspondence between HT bits and hash values (ETHAnHT1 and ETHAnHT2)"*), a receive packet that has that multicast address will be accepted. To set the hash table, it is necessary to execute a CRC calculation on a multicast address defined in advance, and set the corresponding bits.

Table 23-116 Correspondence between HT bits and hash values (ETHAnHT1 and ETHAnHT2)

	CRC[25:23]							
CRC[28:26]	**111_B**	**110_B**	**101_B**	**100_B**	**011_B**	**010_B**	**001_B**	**000_B**
111_B	HT1[31]	HT1[30]	HT1[29]	HT1[28]	HT1[27]	HT1[26]	HT1[25]	HT1[24]
110_B	HT1[23]	HT1[22]	HT1[21]	HT1[20]	HT1[19]	HT1[18]	HT1[17]	HT1[16]
101_B	HT1[15]	HT1[14]	HT1[13]	HT1[12]	HT1[11]	HT1[10]	HT1[9]	HT1[8]
100_B	HT1[7]	HT1[6]	HT1[5]	HT1[4]	HT1[3]	HT1[2]	HT1[1]	HT1[0]
011_B	HT2[31]	HT2[30]	HT2[29]	HT2[28]	HT2[27]	HT2[26]	HT2[25]	HT2[24]
010_B	HT2[23]	HT2[22]	HT2[21]	HT2[20]	HT2[19]	HT2[18]	HT2[17]	HT2[16]
001_B	HT2[15]	HT2[14]	HT2[13]	HT2[12]	HT2[11]	HT2[10]	HT2[9]	HT2[8]
000_B	HT2[7]	HT2[6]	HT2[5]	HT2[4]	HT2[3]	HT2[2]	HT2[1]	HT2[0]

An example of a program that executes hash table calculations is shown below.

If DA = 123456789ABC, for example, CRC = D4E88056, CRC[28:26] = 5, and CRC[25:23] = 1. If HT1[9] in *Table 23-116 "Correspondence between HT bits and hash values (ETHAnHT1 and ETHAnHT2)"* is set, a multicast packet with the target DA is received. If the value of both the ETHAnHT1 and ETHAnHT2 registers is 00000000H, all packets are discarded.

// Calculate the set value of the hash table.

```
#include <stdio.h>

unsigned long crc32_for_ethernet( const unsigned char *data, int size );

// Address to be calculated
const unsigned char DA[] = { 0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC };

int main( void ){
    unsigned long crc;

    printf("´nDA: ");
    crc = crc32_for_ethernet( DA, sizeof(DA) );
    printf("-----------------------------------------´n");
    printf("CRC = %02X,%02X,%02X,%02X´n", (crc>>24)&0xff, (crc>>16)&0xff, (crc>>8)&0xff,
    crc&0xff );
    printf("CRC[28:26] = %X, CRC[25:23] = %X ´n", (crc>>26)&0x07, (crc>>23)&0x07 );
    printf("´n");
    return(1);
    }

// Calculate the CRC.
unsigned long crc32_for_ethernet( const unsigned char *p, int size ){
    int i,j;
    const unsigned long poly = 0xEDB88320ul; // BigEndian
    unsigned long crc = 0xffffffff;
    unsigned long ans = 0x00000000;
    unsigned char c;

    for( j = 0; size-- != 0 ; j++ ) {
        c = *p++;
        printf("%02X " , c );
        if ( j == 15 ) {
            j = 0;
            printf("´n");
            }
        for ( i = 0; i < 8; i++ ) {
            crc = (crc>>1)^(((crc^c)&1)? poly : 0ul );
            c >>= 1;
            }
        }
    if ( j != 0 ) printf("´n");
    crc = ~crc;
    for( i = 0; i < 4; i++ ){
        ans = (ans << 8) | (crc & 0x000000ffful);
        crc >>= 8;
        }
    return( ans );
}
```

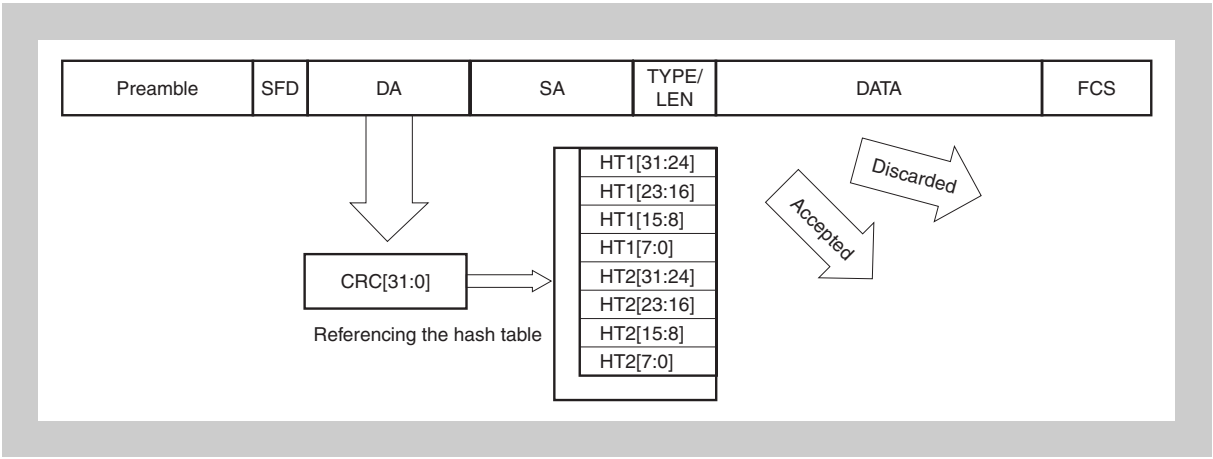


Figure 23-18 Image of filtering by referencing hash table

(c) Filtering of broadcast addresses

When the ETHAnAFR.ABC bit is set, a packet that has a broadcast address is received.

(d) Promiscuous mode

When the ETHAnAFR.PRO bit is set, the promiscuous mode is set and all packets are received.

If none of reception conditions (a) to (d) above is satisfied, the received packet is discarded.

For the combinations of the above conditions, see *Table 23-117 "Settings of ETHAnAFR register and packets to be filtered"*.

Table 23-117 Settings of ETHAnAFR register and packets to be filtered

Setting of ETHAnAFR register				Receive packet				
PRO	PRM	AMC	ABC	LSA mismatch, unicast	LSA match, unicast	HT mismatch, multicast	HT match, multicast	Broadcast packet
1	–	–	–	Accepted	Accepted	Accepted	Accepted	Accepted
0	1	–	–	Discarded				
0	0	1	1			Discarded		
0	0	1	0					Discarded[a]
0	0	0	1				Discarded	Accepted
0	0	0	0					Discarded

a) The broadcast packet can be received if the corresponding bit in the hash table is set because the broadcast address is included in a multicast address.

Note –: Any

(2) Setting address filtering conditions

Set address filtering as follows.

First, clear the ETHAnMACC1.SRXEN bit. When the SRXEN bit is 0, the receive data interface is disabled. Next, set a station address to the ETHAnLSA1 and ETHAnLSA2 registers. Set a combination of the necessary filtering conditions to the ETHAnAFR register. To conditionally receive multicast packets, a hash table must be set up by using the ETHAnHT1 and ETHAnHT2 registers. After making the above settings, enable packet reception by setting the ETHAnMACC1.SRXEN bit.

23.5.8 Statistics counters

The Ethernet controller has 39 statistics counters to check the communication quality and other line statuses.

Each time communication of one frame has been finished (or aborted), the communication status is checked and the corresponding statistics counter is updated. The statistics counters cannot be stopped. To avoid using a statistics counter, set the corresponding bit in the ETHAnCAM1 or ETHAnCAM2 register to mask the interrupt from that counter.

The statistics counters can be read at any time even during communication.

If a counter overflows, the corresponding bit in the ETHAnCAR1 or ETHAnCAR2 register is set, and, if the relevant interrupt is not masked by the ETHAnCAM1 and ETHAnCAM2 registers, an Ethernet MAC interrupt is generated. The ETHAnCAM1 and ETHAnCAM2 registers can be used to specify whether to mask the overflow interrupt of each counter.

To clear a statistics counter, write 0 to it. At this time, the current communication does not have to be stopped. If updating a statistics counter and writing data to it occur at the same time and conflict, updating takes precedence, and the counter is written after it has been updated.

Note that the statistics counters cannot be stopped. To avoid using a statistics counter, mask the counter by setting the corresponding bit in carry mask register 1 or 2 (ETHAnCAM1 and ETHAnCAM2) to prevent the INTETMOV interrupt from being generated.

The statistics counters can be read or written in 32-bit units.

Cautions

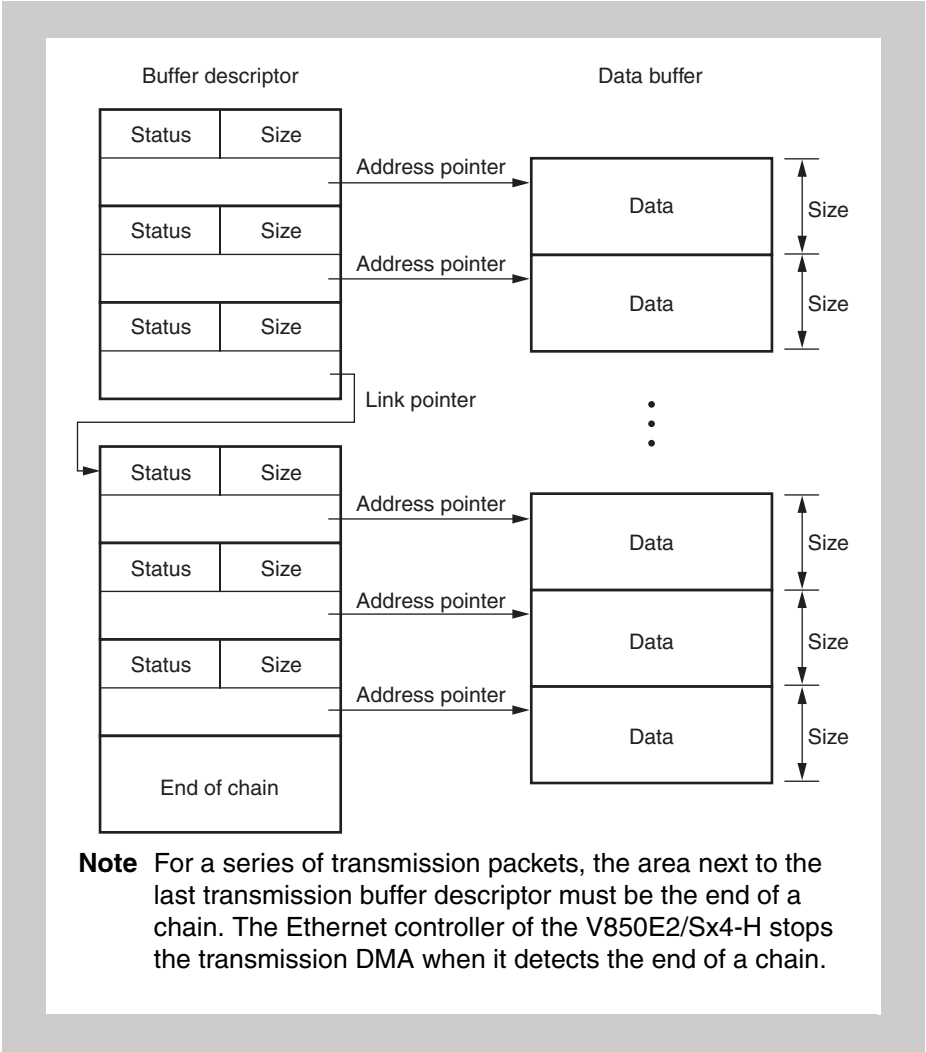
1. The Ethernet controller updates the statistics counters based on the Ethernet controller clock (f_{EC}). If the Ethernet controller clock (f_{EC}) is considerably slower than the communication clock (TXCLK/RXCLK), the counters might miscount statistics information. If the statistics information is miscounted, a status vector overrun occurs, the C2DV bit in carry register 2 (ETHAnCAR2) is set, and an INTETMOV interrupt is generated.
2. Carry registers 1 and 2 (ETHAnCAR1 and ETHAnCAR2) are cleared when they are read.

Note Aside from the statistics counters, a transmission abort counter (ETHAnTXABTCNT) and a reception abort counter (ETHAnRXABTCNT) are available for counting the number of times transmission/reception has been aborted.

23.6 Data Transfer

23.6.1 Buffer structure

The Ethernet controller buffer of the V850E2/Sx4-H consists of a buffer descriptor and data buffer.



Note For a series of transmission packets, the area next to the last transmission buffer descriptor must be the end of a chain. The Ethernet controller of the V850E2/Sx4-H stops the transmission DMA when it detects the end of a chain.

Figure 23-19 Ethernet controller buffer structure (transmission)

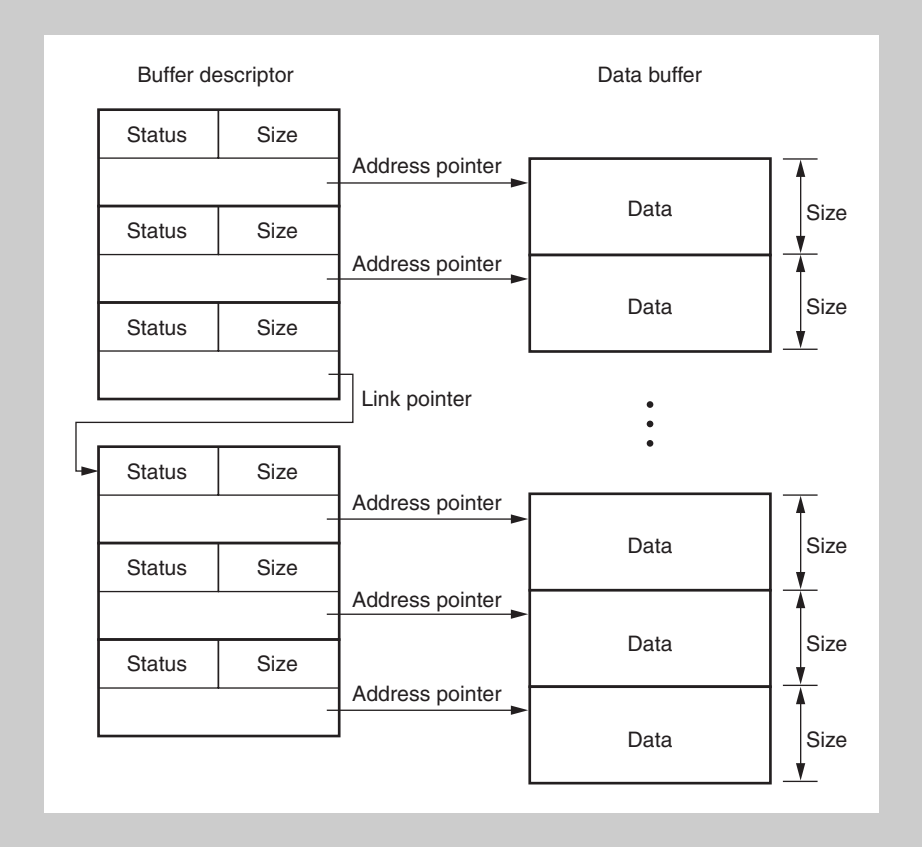


Figure 23-20 Ethernet controller buffer structure (reception)

23.6.2 Descriptor mechanism

The Ethernet controller supports a descriptor mechanism to support a situation where the memory space that stores transmit data and receive data is not contiguous.

The Ethernet controller uses the following three types of descriptors.

- Buffer descriptor
- Link pointer
- End of chain

Each descriptor consists of two word-aligned data (64 bits).

The Ethernet controller can consecutively process multiple descriptors in one DMA transfer (see *(6) "Descriptor chain" on page 1770 in 23.6.2 "Descriptor mechanism"*).

A reception DMA transfer or transmission DMA transfer is started by setting the first address of a receive descriptor chain to ETHAnRXDP or the first address of a transmit descriptor chain to ETHAnTXDP, and then setting the RXS or TXS bit of the ETHAnMODE register.

A descriptor chain must end with the descriptor of an end of chain.

(1) Format of buffer descriptor

A buffer descriptor is configured of 2 words (64 bits). The lower word consists of control bits. The higher word indicates the start address of the data buffer indicated by the descriptor.

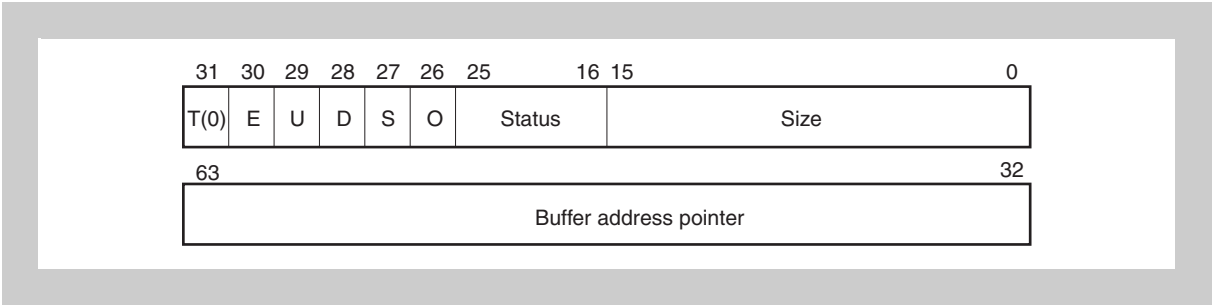


Figure 23-21 Buffer descriptor format

Table 23-118 Bit configuration of buffer descriptor (1/2)

Bit position	Bit name	Function
63 to 32	BAP	These bits specify an address pointer that indicates the start address of the data buffer. Byte alignment can be specified for BAP.
31	T	Descriptor type This bit indicates the type of the descriptor. Clear this bit for a buffer descriptor.
30	E	Last buffer flag This is a control bit that indicates the end of packet data. 0: Normal buffer data (not last data) 1: Last buffer data of the current packet If this bit is set for transmission, a TXI interrupt is generated when the data of the corresponding data buffer has been transferred, and then the next descriptor processing starts. Clear this bit for reception. When the last data of a frame has been written, this bit is set when the data is written back. Next, an RXI interrupt is generated, and then the next descriptor processing starts.
29	U	Used bit This bit indicates whether a DMA transfer has finished (including transfer in progress). 0: Transfer not finished (including transfer in progress) 1: Transfer finished The CPU clears this bit when it creates or obtains buffer data (a descriptor). When a DMA transfer to the buffer area indicated by this descriptor has finished, the Ethernet controller sets this bit. The Ethernet controller issues the TECI or RECI interrupt and stops DMA if it reads a descriptor whose U bit is set. If a bus error or overflow error occurs during reception, the U bit of either the first descriptor in the packet or the descriptor where the overflow occurred is set.
28	D	This bit indicates an access error in the data buffer. 0: No error 1: Access error in the data buffer The CPU clears this bit when it creates or obtains buffer data (a descriptor). If an access error occurs, the Ethernet controller sets control bit D of the first descriptor that indicates the current packet, and control bit D of the descriptor that was responsible for the access error.
27	S[a]	This bit indicates that reception status information has been written to the Status field (only control bit S in the first descriptor of a received packet is valid). 0: Status information is not included. 1: Status information of the received packet is included. The CPU clears this bit when it creates or obtains buffer data (a descriptor). When a received packet is transferred via DMA, the Ethernet controller writes a valid value to the Status field of the first descriptor of the current packet and sets control bit S each time one packet has been transferred.

Table 23-118 Bit configuration of buffer descriptor (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>26</td><td>O[a]</td><td>This bit indicates occurrence of an overflow error during reception.
0: No overflow
1: Overflow
The CPU clears this bit when it creates or obtains buffer data (a descriptor). If an overflow error occurs during reception, the Ethernet controller writes back 1 to the control bit O of the first descriptor of the packet, and sets control bit E of the descriptor in which the overflow error occurred. No interrupt is generated.</td></tr>
<tr><td>25:16</td><td>Status[a]</td><td>These bits form a Status field that indicates status information during reception. If control bit S is 1, the value of the Status field is valid. The CPU clears this bit when it creates or obtains buffer data (a descriptor). During a DMA transfer of a receive packet, the Ethernet controller writes a valid value to the Status field of the first descriptor of the current packet and sets control bit S each time transfer of one packet has finished.
<table>
<tr><th>Bit position</th><th colspan="2">Bit name</th></tr>
<tr><td>16</td><td colspan="2">CEPS</td></tr>
<tr><td>17</td><td colspan="2">RCV</td></tr>
<tr><td>18</td><td colspan="2">RCRCE</td></tr>
<tr><td>19</td><td colspan="2">RLOR</td></tr>
<tr><td>20</td><td colspan="2">DBNB</td></tr>
<tr><td>21</td><td colspan="2">RXOK</td></tr>
<tr><td>23:25</td><td>FYTP[0:2]</td><td>000: RBRO
001: RMUL
010: USOP
011: VLAN
100: RPCF
101: RCFR
110: “Normal”
111: “Reserved”</td></tr>
</table></td></tr>
<tr><td>15 to 0</td><td>Size</td><td>These bits form a Size field that indicates the size (in bytes) of the buffer data indicated by this descriptor. During a DMA transfer of a receive packet, the Ethernet controller writes the length of one transferred packet to the Size field of the last descriptor of the current packet each time transfer of one packet has finished.</td></tr>
</table>

a) This bit is not used during transmission. Clear this bit.

Note The Size field is 16 bits. Setting 0 to this field is prohibited. If 0 is set, an error interrupt is generated.
If FFFFH is set to this field, transfer of 64K – 1 bytes is executed.

(2) Format of link pointer

A link pointer consists of 2 words. The lower word consists of control bits. The higher word indicates the address of the next descriptor.

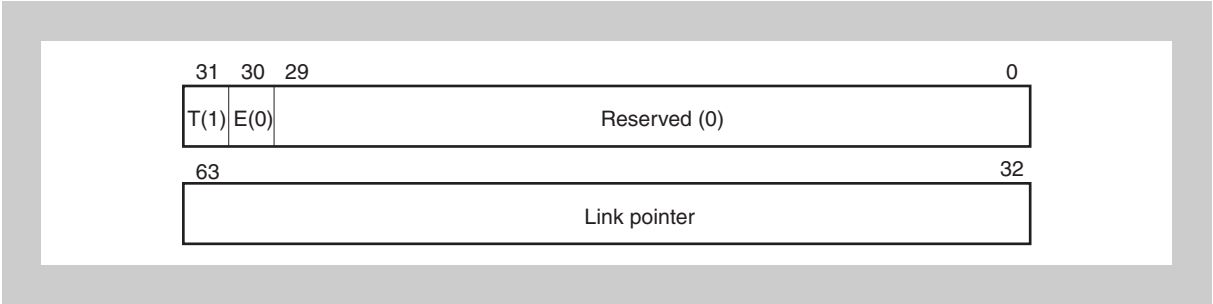


Figure 23-22 Link pointer format

Table 23-119 Bit configuration of link pointer

Bit position	Bit name	Function
63:32	Link Pointer	These bits indicate the address of the next descriptor. The lower 2 bits are ignored (word aligned).
31	T	Set this bit for a link pointer.
30	E	Clear this bit for a link pointer.
29:0	Reserved	These bits are reserved. Clear these bits.

(3) Format of end of chain

An end of chain consists of 2 words. The lower word consists of control bits. The higher word indicates 0.

When the Ethernet controller detects an end of chain, it finishes the DMA transfer and generates an RECI or TECI interrupt.

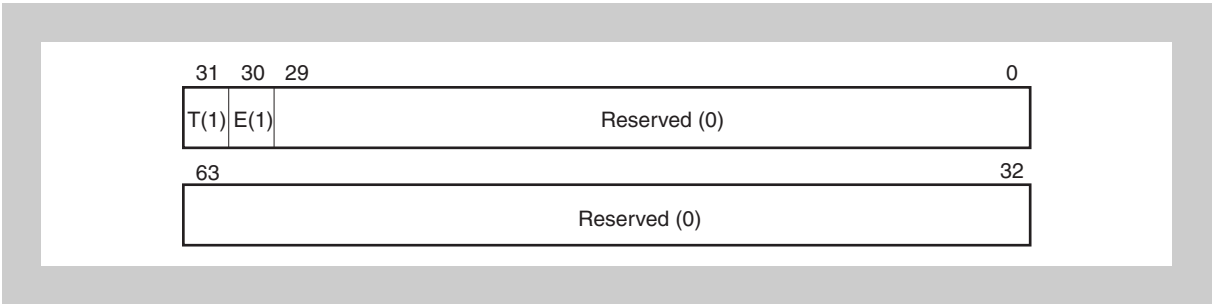


Figure 23-23 End of chain format

Table 23-120 Bit configuration of end of chain

Bit position	Bit name	Function
63:32	BAP	These bits are set to null (all zero) for an end of chain.
31	T	Set this bit for an end of chain.
30	E	Set this bit for an end of chain.
29:0	Reserved	These bits are reserved. Clear these bits.

(4) Writing back the status

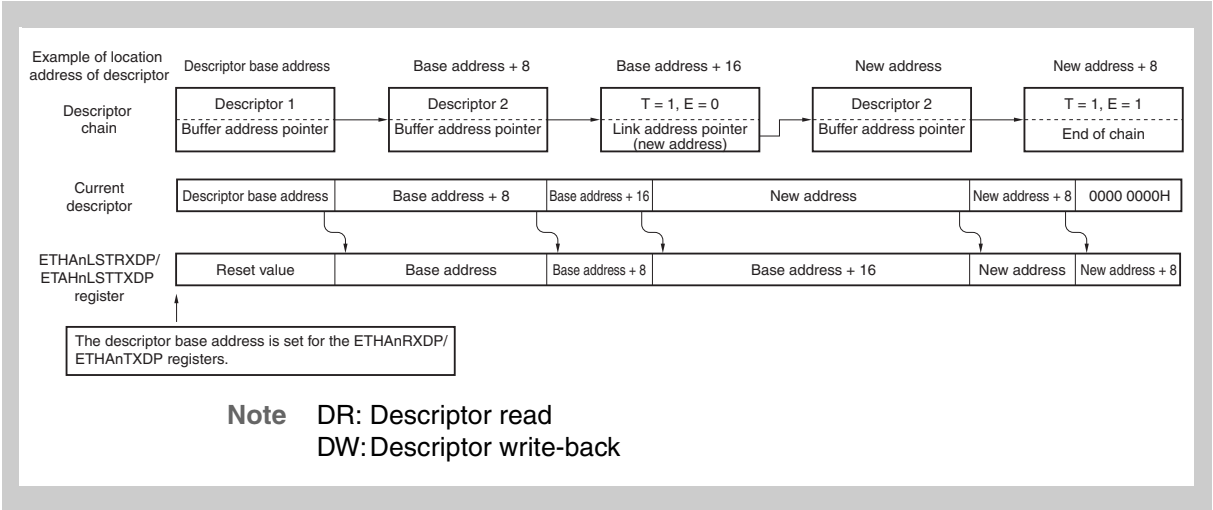
When DMA reception is executed, the reception status is written back to the first descriptor of the packet, and the length of the packet transferred via DMA is written back to the last descriptor. The status is written back as described in the *Status* field in *Table 23-118 "Bit configuration of buffer descriptor"*.

(5) Reporting last descriptor

The current descriptor can be reported. Two registers, ETHAnLSTRXDP and ETHAnLSTTXDP, hold the address of the descriptors processed by the Ethernet controller. The address of the descriptor that was processed immediately before can be obtained by reading these two registers via software.

The timing to save the address of a descriptor to ETHAnLSTRXDP or ETHAnLSTTXDP is shown in *Figure 23-24 "Timing to copy last descriptor"* below.

When the link pointer is read, the address of the next descriptor can be read from the BAP bits. The address of the link pointer is then copied to ETHAnLSTRXDP or ETHAnLSTTXDP.



Note DR: Descriptor read
DW: Descriptor write-back

Figure 23-24 Timing to copy last descriptor

If a descriptor chain is configured in a ring buffer, the descriptor can be updated by reading ETHAnLSTRXDP or ETHAnLSTTXDP, using the TXI flag of the INTSCTX interrupt (the RXI flag of the INTSCRX interrupt) as a trigger.

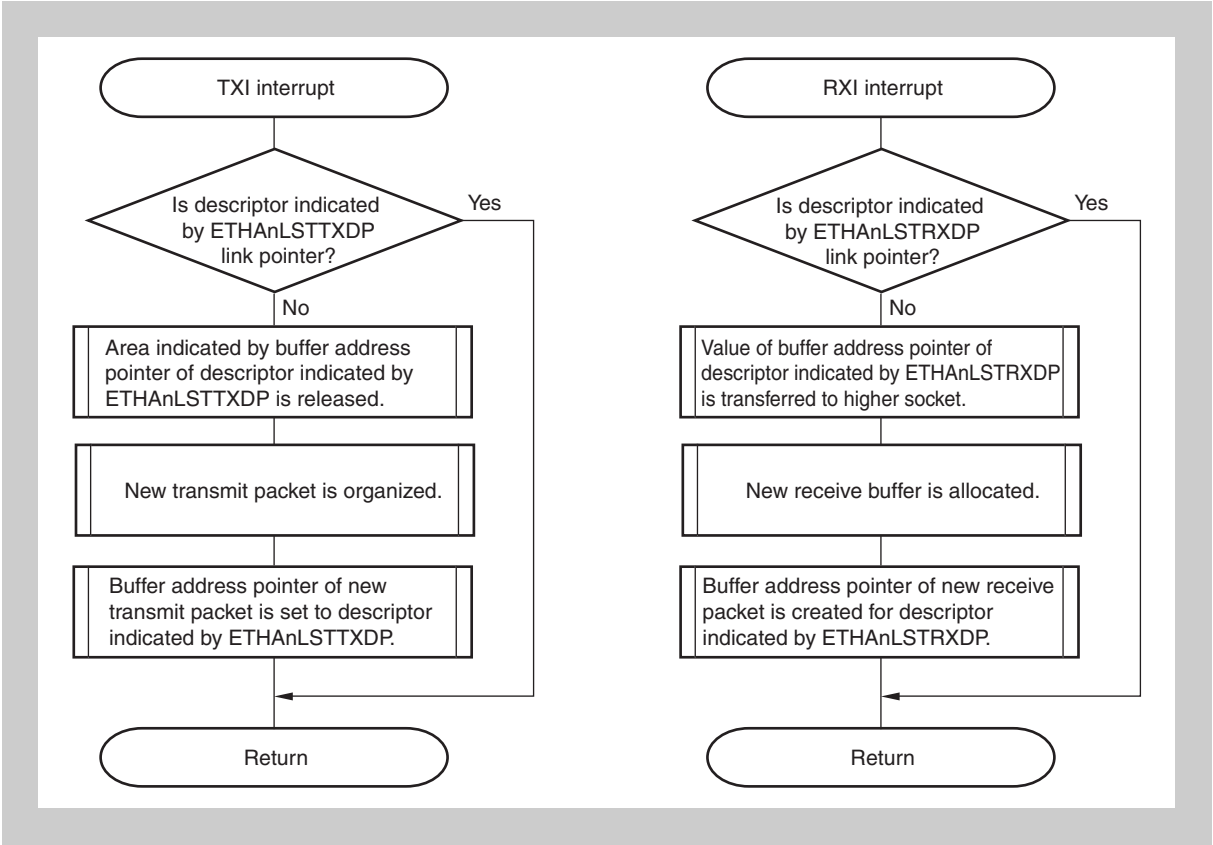


Figure 23-25 Updating descriptor chain by using ETHAnLSTRXDP or ETHAnLSTTXDP

(6) Descriptor chain

A descriptor uses a chain structure to indicate a data buffer (of an undefined length).

An image is as shown below.

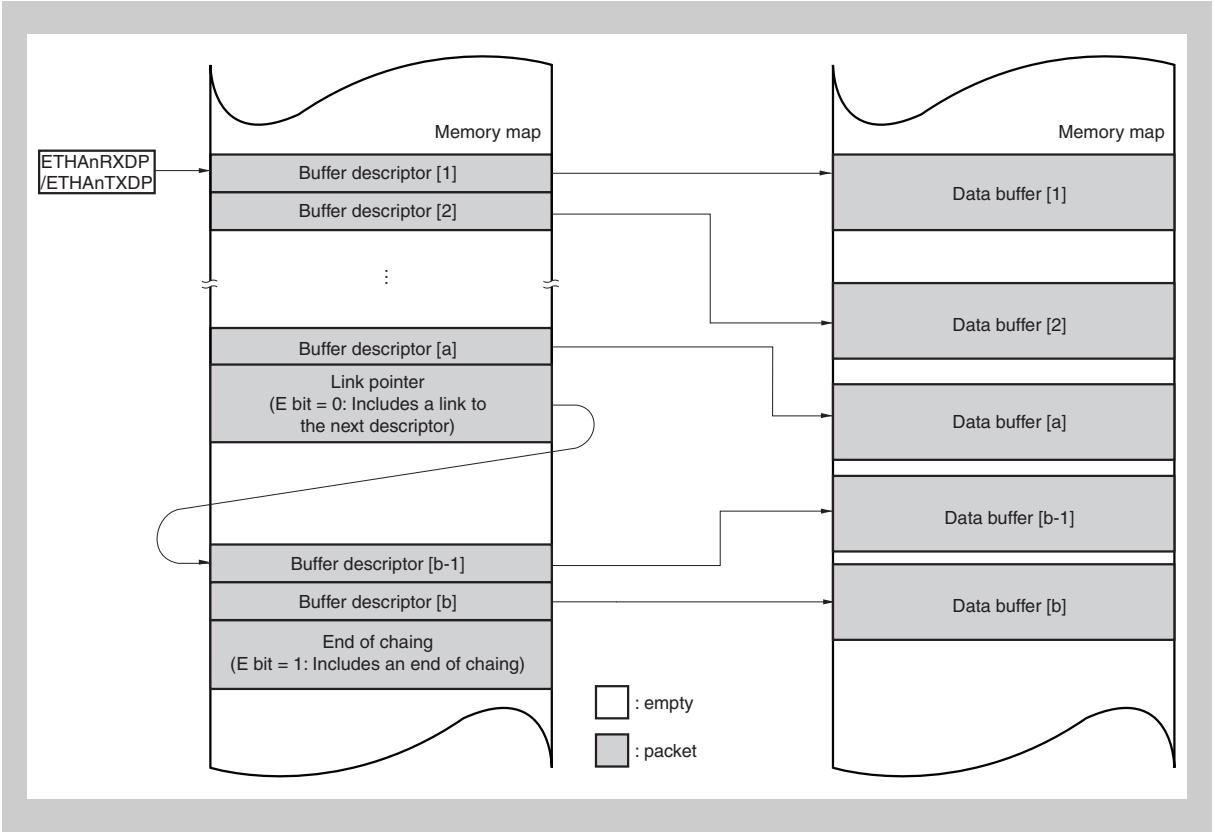


Figure 23-26 Overview of descriptor chain using ETHAnLSTRXDP or ETHAnLSTTXDP

Allocate a descriptor to a successive memory area. When a link pointer is used, a descriptor can also be allocated to a non-successive memory area by specifying the location address of the next buffer descriptor for the link pointer. The descriptor chain can be finished by specifying the end of a chain.

A ring descriptor chain may be configured by using the last link pointer to specify the memory address of the first buffer descriptor.

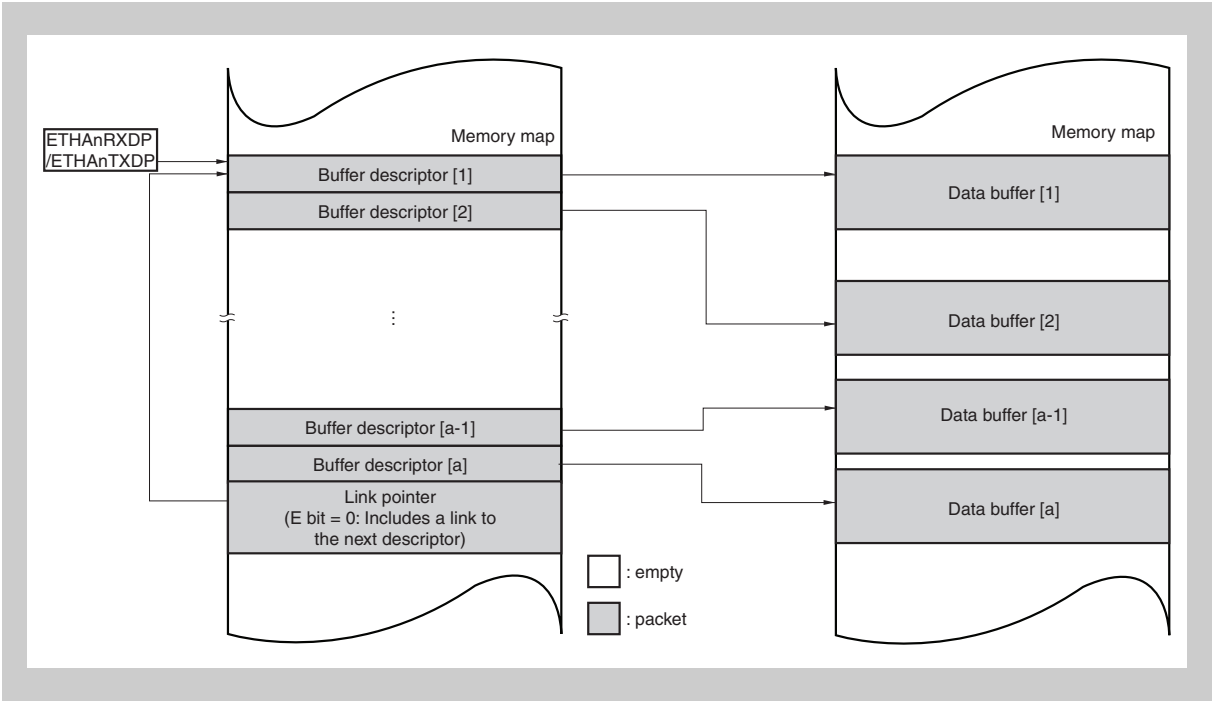


Figure 23-27 Overview of ring buffer formed by descriptor chain

A ring buffer is configured if the beginning of a descriptor chain is specified by a link pointer. In a ring buffer, if a descriptor whose U bit is set is read, the Ethernet controller generates a RECI or TECI interrupt in the same manner as when detecting the end of a chain, and then stops DMA.

Caution Handling of U bit

If the U bit of a transmit descriptor is set, it indicates that the processing for that descriptor has finished, so the CPU can specify a new descriptor by clearing the U bit.

For a receive descriptor that has the U bit set to 1, however, its content might be updated later due to status write back or error occurrence, so a new descriptor cannot be specified unless the completion of packet reception is confirmed. But if the E bit is set, it indicates that packet reception has finished, so the descriptor chain can be specified as a new descriptor.

(7) Byte alignment and word boundary

Although a descriptor must be word-aligned, the data buffer can be set at a byte-aligned address.

The Ethernet controller automatically identifies an address, executes a single transfer up to a word boundary and an undefined length word transfer up to the burst boundary, and then executes a burst transfer.

23.6.3 Frame transmission

When the CPU prepares the transmission descriptor and data for the transfer target of the DMAC for the Ethernet controller, the transmission descriptor register (ETHAnTXDP) is set up, and then the TXS bit of the ETHAnMODE register is set, the dedicated DMAC fetches the transmission buffer descriptor from the address specified for the descriptor register, reads the transmission data from the data buffer, and then transfers it to the transmission FIFO.

The data transferred to the FIFO buffer is then output to PHY in synchronization with TXCLK, in the order of the preamble, SFD, and then frame data.

If the CRCEN bit of the ETHAnMACC1 register is set, FCS is added to the end of the data.

If the PADEN bit of the ETHAnMACC1 register is set, short frames are automatically padded with 0s when they are transmitted.

If the current descriptor does not contain the end of the frame, the next descriptor is read and data is read from the data buffer indicated by the read descriptor.

When the transmission finishes, the transmission status is written to the last descriptor. Then, the next transmission buffer descriptor is fetched, and the transmission resumes in the same manner when the next data can be transmitted.

Each time DMA processing for a buffer descriptor finishes, an interrupt that indicates the end of transmission DMA (TXI) is generated.

If the next transmission buffer descriptor is the end of a chain descriptor, an interrupt that indicates this (TECI) is generated and transmission DMA stops.

To resume transmission DMA, set up the ETHAnTXDP register and buffer descriptors again.

The frame transmission procedure is described below.

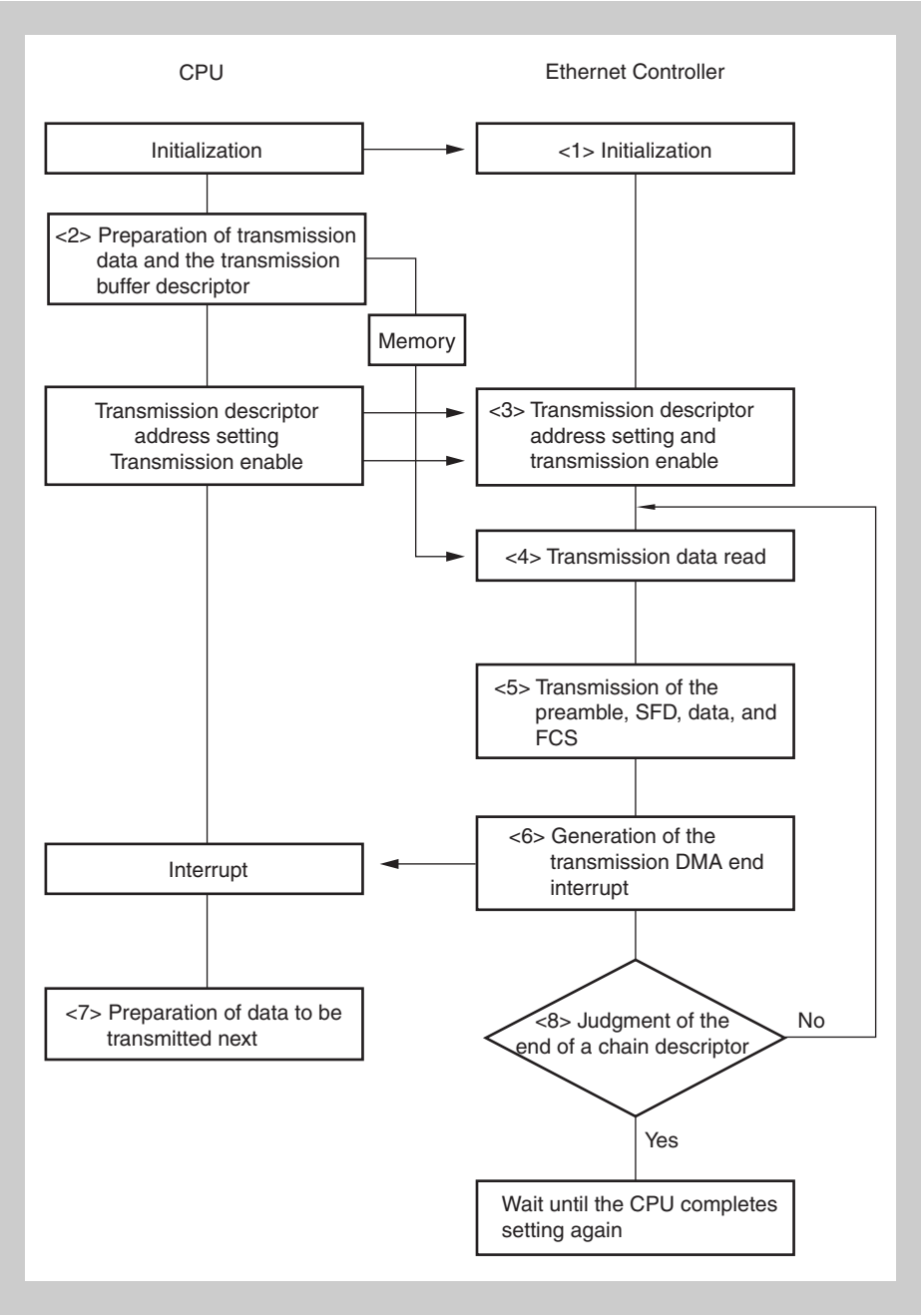


Figure 23-28 Frame transmission procedure example

1. Initialize the Ethernet controller.
   For details about the initialization procedure, see *23.3 "Initialization" on page 1608*.
2. Create transmission data and the transmission buffer descriptor in the data RAM.

   When creating the descriptor, specify values for the E bit (set to 1 if the last packet data needs to be kept), T bit (cleared to 0), U bit (cleared to 0), and Size bits of the transmission buffer descriptor.
3. Set up the transmission buffer descriptor register and enable transmission.

   Specify the transmission descriptor address for the ETHAnTXDP register, and set the TXS bit of the ETHAnMODE register to enable transmission.
4. Read the transmission data.

   The transmission data is read from memory by way of DMA.
   The next descriptor is read if the E bit of the transmission buffer descriptor is 0.
5. Transmit a packet.

   The preamble, SFD, data, and FCS are transmitted.
6. Report the end of DMA transmission.

   A TXI interrupt request is generated to report the end of transmission to the CPU.
7. Prepare the next data.

   The CPU checks the transmission status and prepares the data to be transmitted next.
8. Judge the end of a chain descriptor.

   A TECI interrupt request is generated to report to the CPU that the end of the chain descriptor has been reached.

The frame transmission operation is described below, using a specific descriptor chain as an example.

When the TXS bit of the core function setting register ETHAnMODE is set by software, the first descriptor is read from the address ($0028\ 0000_H$) specified by the transmission descriptor pointer ETHAnTXDP, and transmission descriptor analysis starts. Specify the buffer address pointer ($0028\ 1000_H$) as the DMA transfer start address, and then transfer the data in the buffer to the transmission FIFO.

Because the E bit of the transmission descriptor is 0, which indicates that it is not the last data, the next buffer descriptor ($0028\ 0008_H$) is read, the buffer address pointer ($0028\ 1800_H$) is specified, and then the data in the buffer is transferred to the transmission FIFO.

If the data in the buffer indicated by the buffer descriptor whose E bit is 1 has been transferred completely, the U bit is set and the transmission processing finishes. At this time, an interrupt request TXI is generated.

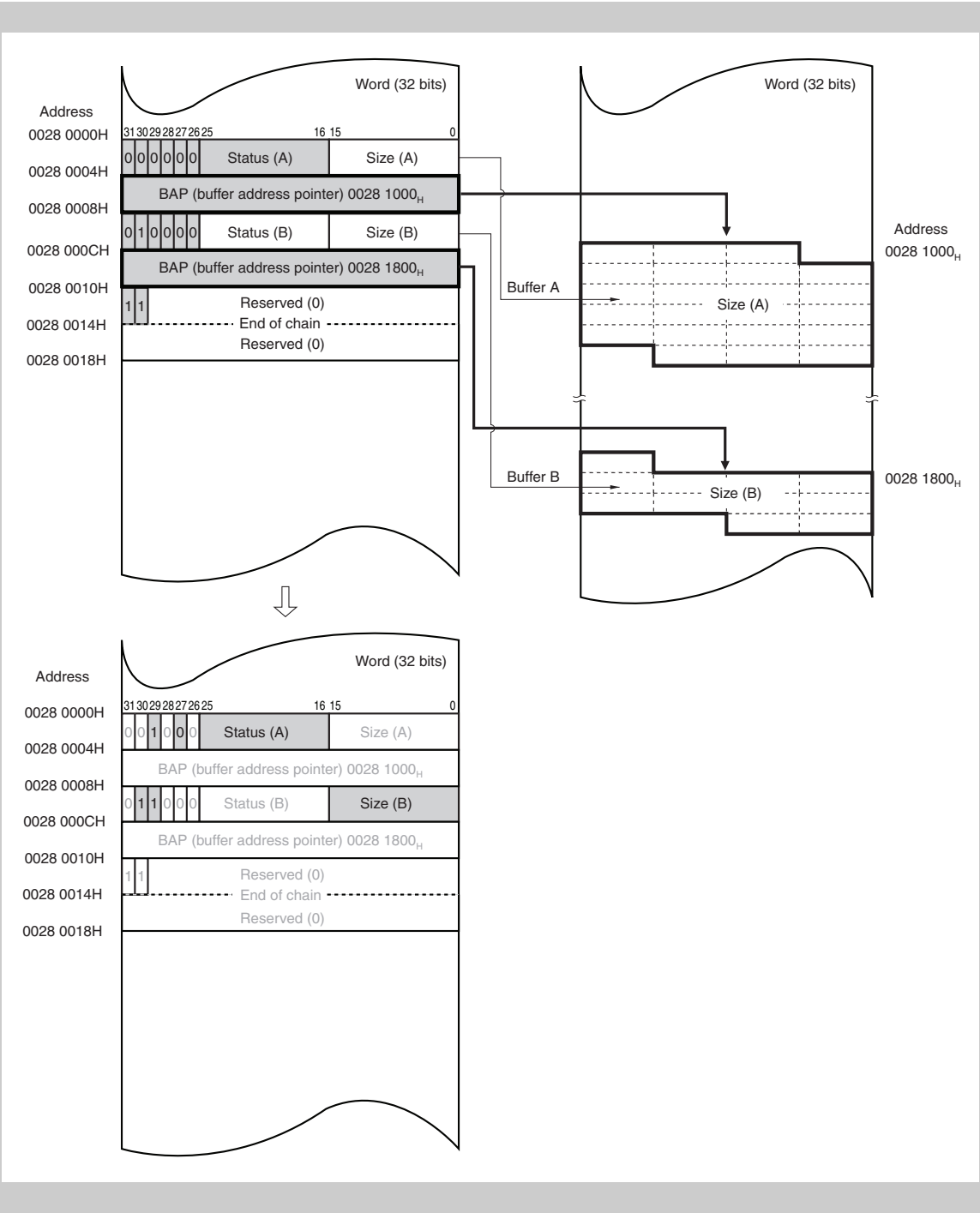


Figure 23-29 Descriptor chain configuration example (packet transmission)

23.6.4 Frame reception

When the of SRXEN bit (reception enable) of the MAC configuration register (ETHAnMACC1) and the RXS bit (reception DMA enable) of the ETHAnMODE register are set and the descriptor pointer register ETHAnRXDP is set up, reception frame processing starts immediately when the MAC receives data.

When data is received, the preamble and frame start delimiter (SFD) are checked to determine whether they are valid.

If they are valid, processing of the received frame starts.

If either is invalid, the frame is ignored.

If a frame conflict occurs or a frame is corrupted due to address filtering, the data is not written to the reception buffer.

When the frame is normally received and not corrupted by address filtering, it is transferred to the data buffer specified by the reception buffer descriptor.

During reception, the Ethernet controller checks whether the frame length is correct.

At the end of the frame, the FCS is checked and written to the buffer descriptor.

A frame of 64 bytes or shorter (a short packet) is DMA transferred.

When the frame has been received completely, the E and U bits of the last descriptor are set and the number of data bytes transferred to the Size field is written back.

When the all packet data has been received, the U and S bits of the first descriptor are set and the reception status is written back to the Status field.

At this time, an interrupt request RXI is generated.

The frame reception procedure is described below.

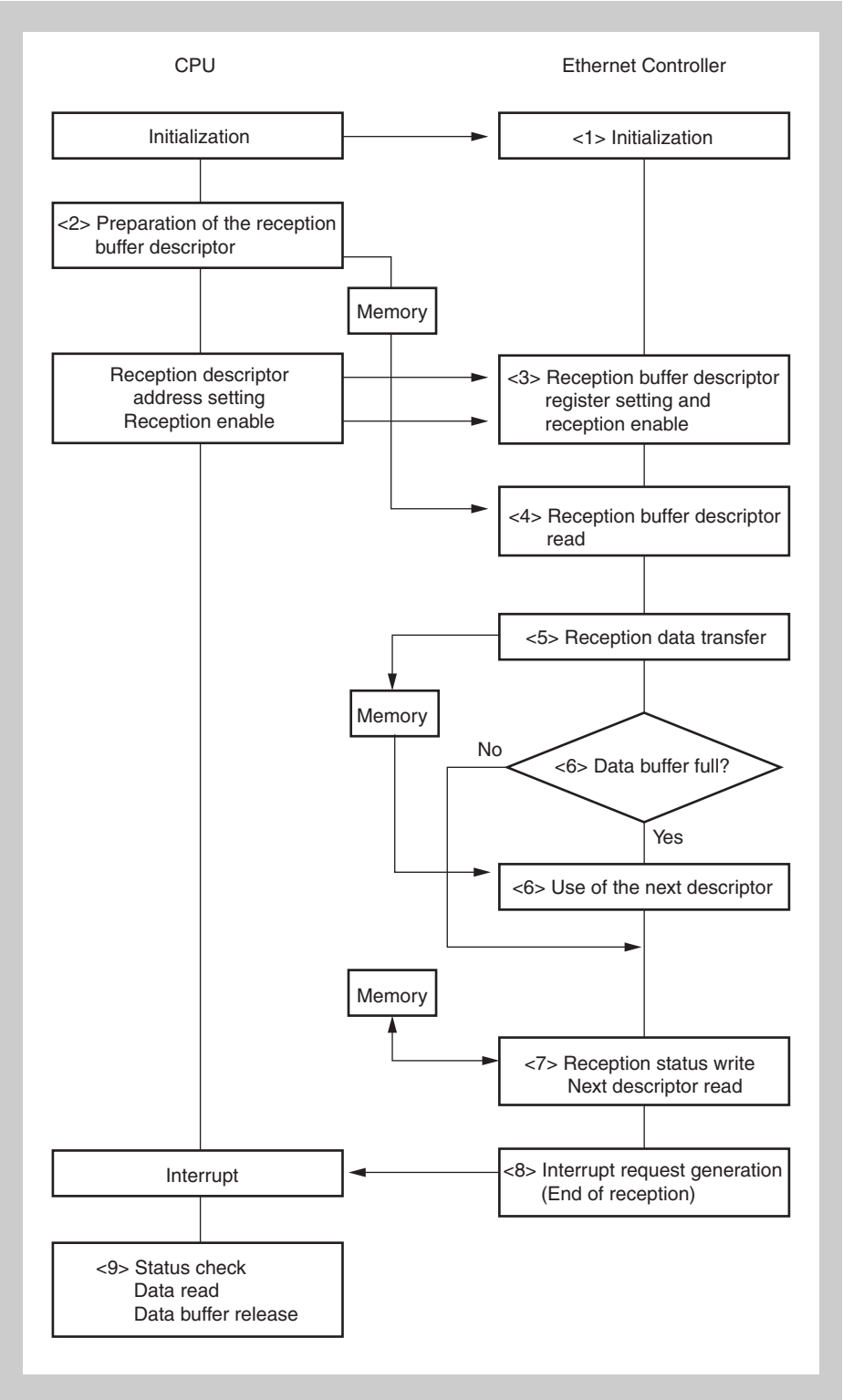


Figure 23-30 Frame reception procedure example

1. Initialize theEthernet controller.
   For details about the initialization procedure, see *23.3 “Initialization” on page 1608*.
2. Creating a reception buffer descriptor in memory.

   When creating the descriptor, specify values for the T bit (cleared to 0), U bit (cleared to 0), and Size bits (data buffer size) of the reception buffer descriptor.
3. Set up the reception buffer descriptor register and enable reception.

   Specify the reception descriptor address for the ETHAnRXDP register, and set the RXS bit of the ETHAnMODE register to enable reception.
4. Read the reception buffer descriptor.

   The reception buffer descriptor is read by way of DMA.
5. Transfer reception data.

   The data is transferred to the data RAM by way of DMA.
6. Judge whether the reception data buffer is full.

   The next descriptor is read if the current data buffer is full.
7. Receive a packet.

   Reading of the buffer descriptor and data transfers are repeatedly performed. When the end of the frame is received, 1 is written to the E bit of the last reception buffer descriptor and the number of bytes of transferred data is written to the Size field.
8. Report the end of reception.

   An RXI interrupt request is generated to report the end of reception to the CPU (when the interrupt is not masked).
9. Prepare the next data.

   The CPU checks the reception status, releases the data buffer, and then prepares the next data.

The frame reception operation is described below, using a specific descriptor chain as an example.

When the RXS bit of the core function setting register ETHAnMODE is set by software, the first descriptor is read from the address ($0028\ 0000_H$) specified by the reception descriptor pointer ETHAnRXDP, and reception descriptor analysis starts.

Specify the first buffer address pointer ($0028\ 1000_H$) as the DMA transfer start address, and then transfer the reception data in the FIFO buffer to buffer A.

When buffer A becomes full during the subsequent reception, the next descriptor ($0028\ 0008_H$) is read, the buffer address pointer ($0028\ 1800_H$) is specified as the DMA transfer start address, and then the reception data in the FIFO buffer is transferred to buffer C.

The E and U bits of the last descriptor are set and the number of bytes of data transferred to the Size field is written back.

When all packet data has been transferred, the U and S bits of the first descriptor are set and the reception status information is written back to the Status (A) field.

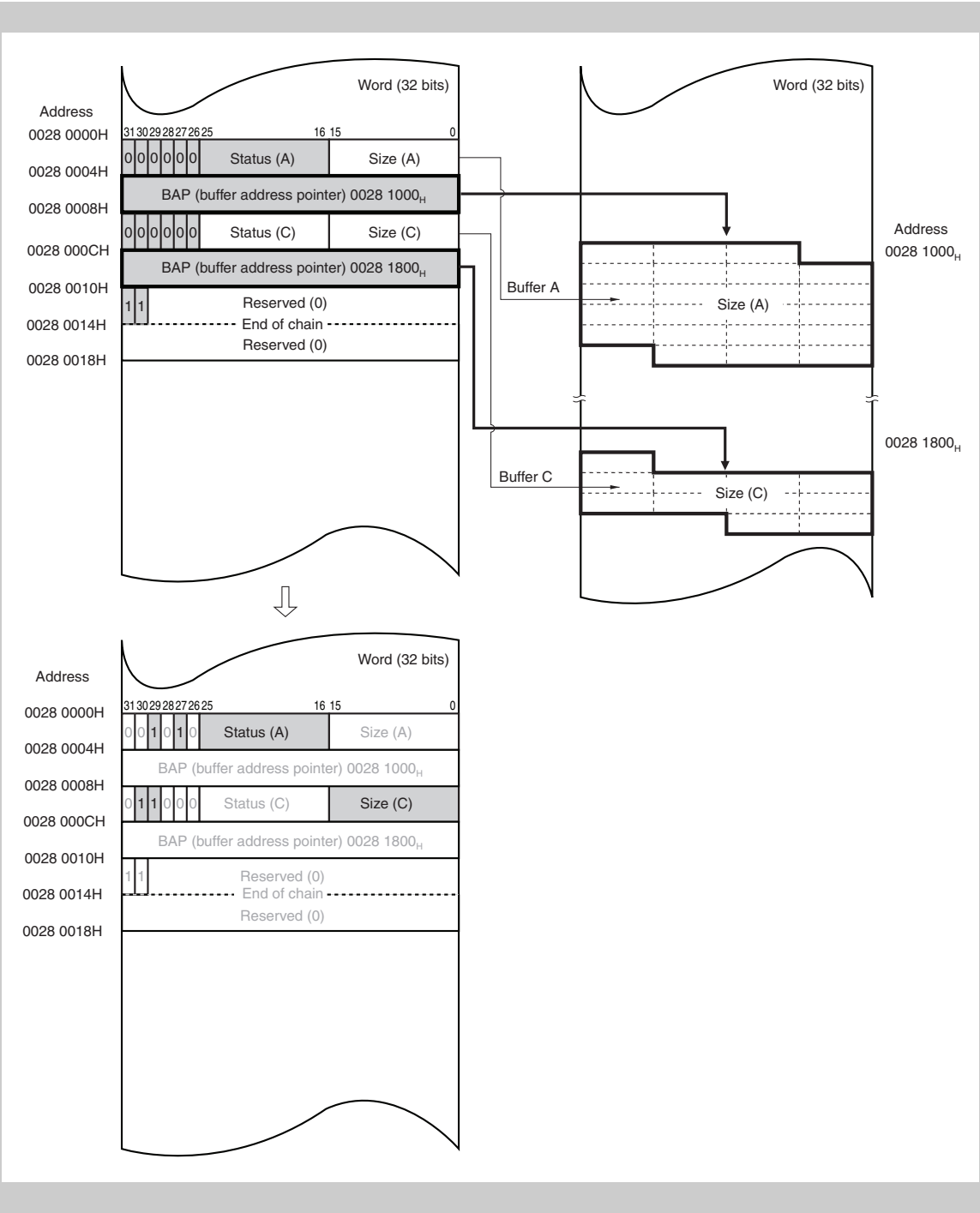


Figure 23-31 Descriptor chain configuration example (packet transmission)

23.6.5 Error processing

(1) Error write back

If a bus error occurs when accessing the data buffer during reception or transmission, an error interrupt is asserted and DMA is stopped. In addition, the U and D bits of the first descriptor in the packet are set (the U bit might already be set), and the U, D, and E bits of the descriptor where the error occurred are set.

If an overflow error occurs during reception, the U and O bits of the first descriptor in the packet are set (the U bit might already be set), and the U and E bits of the descriptor where the error occurred are set.

(2) Error interrupt

An error interrupt is not asserted only by a data buffer access error but also by a descriptor access error. Whether an error interrupt is generated can be checked using the ETHAnINTMS.RBEI and ETHAnINTMS.TBEI bits.

If a data buffer access error or descriptor access error has occurred, the descriptor chain that includes the descriptor where the error occurred must be set up again.

During transmission:

If a data buffer access error or descriptor access error has occurred, the TBEI bit is set and the DMA is stopped.
Transmission will not be performed until the TXS bit is set again.

During reception:

If a data buffer access error or descriptor access error has occurred, the RBEI bit is set and the DMA is stopped.
Reception will not be performed until the RXS bit is set again.
At this time, the packet being transferred from the FIFO buffer is canceled.
Packet cancellation is not executed if no packet is being transferred.
If a bus error occurred during write back due to a reception overflow, the processing is performed in the same way as when a descriptor access error occurs.

23.7 Receive Checksum

The Ethernet controller has a receive checksum unit that calculates the receive checksum.

Receive checksum calculation is enabled and disabled by setting the RXCHKSMEN bit of the ETHAnTRANSCTL register.

A checksum is appended to the end of receive data. Allocate enough memory for the amount of received data, which is the packet length plus 2 bytes.

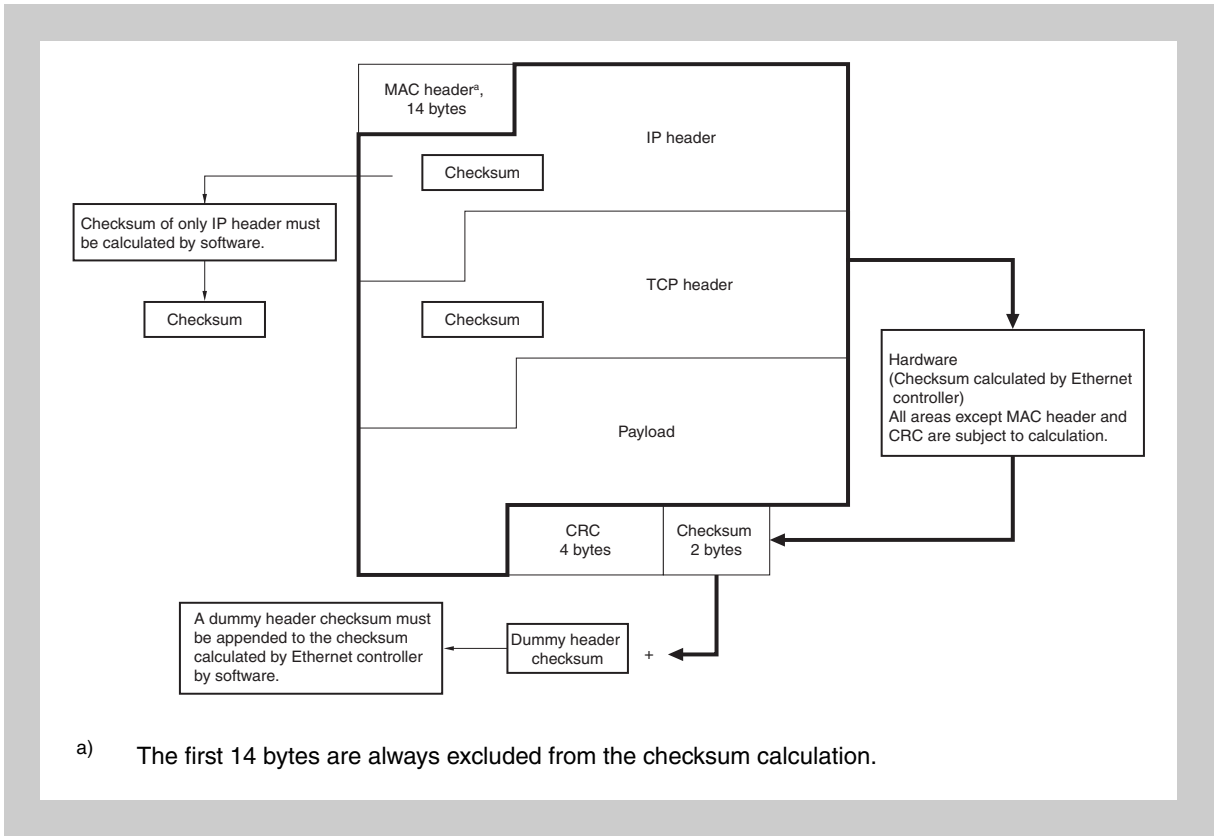
When checksum calculation is enabled, all parts (payload) of a receive frame, except the MAC header (first 14 bytes) and CRC (last 4 bytes), are subject to checksum calculation. If the number of bytes subject to calculation is odd, 00H is added to the last byte for checksum calculation.

The minimum receive packet length subject to checksum calculation is 19 bytes (payload = 1 byte). If the receive packet length is 18 bytes (payload = 0 bytes) or less, 0 is output as the checksum. However, the length information increases by 2 bytes.

Before changing the value of the RXCHKSMEN bit, confirm that transfer of the receive frame is stopped.

23.7.1 Processing by software

The first 14 bytes of a packet are treated as the MAC header and are always excluded from checksum calculations. Therefore, calculation starts from the 15th byte. If the MAC header exceeds 14 bytes, such as in a VLAN and huge frame, correction by software is necessary.



a) The first 14 bytes are always excluded from the checksum calculation.

Figure 23-32 Checksum calculation

The minimum packet length is 64 bytes. When padding a short packet, many systems use 00_H, but some systems use a different code for padding. In this case, the checksum calculation is executed including the padding data, resulting in a mismatch between the calculation result and the checksum in the header. In this case, the checksum of the padding data must be corrected by software.

Caution According to RFC1071, the correct checksum comparison result can be obtained if the checksum result is compared using the same endian format (either big endian or little endian).

23.8 Transmit Checksum

The Ethernet controller has a transmit checksum function. This function has a dedicated DMAC (transmit checksum DMAC), which accesses the internal system bus independently of the internal Ethernet controller DMAC.

The algorithm for a transmit checksum complies with RFC1071, a payload is added in a 32-bit width, and a 2-byte checksum is output to the end of the data.

The minimum transmit payload subject to checksum calculations is 1 byte, and the maximum is 1,500 bytes as defined in IEEE802.3.

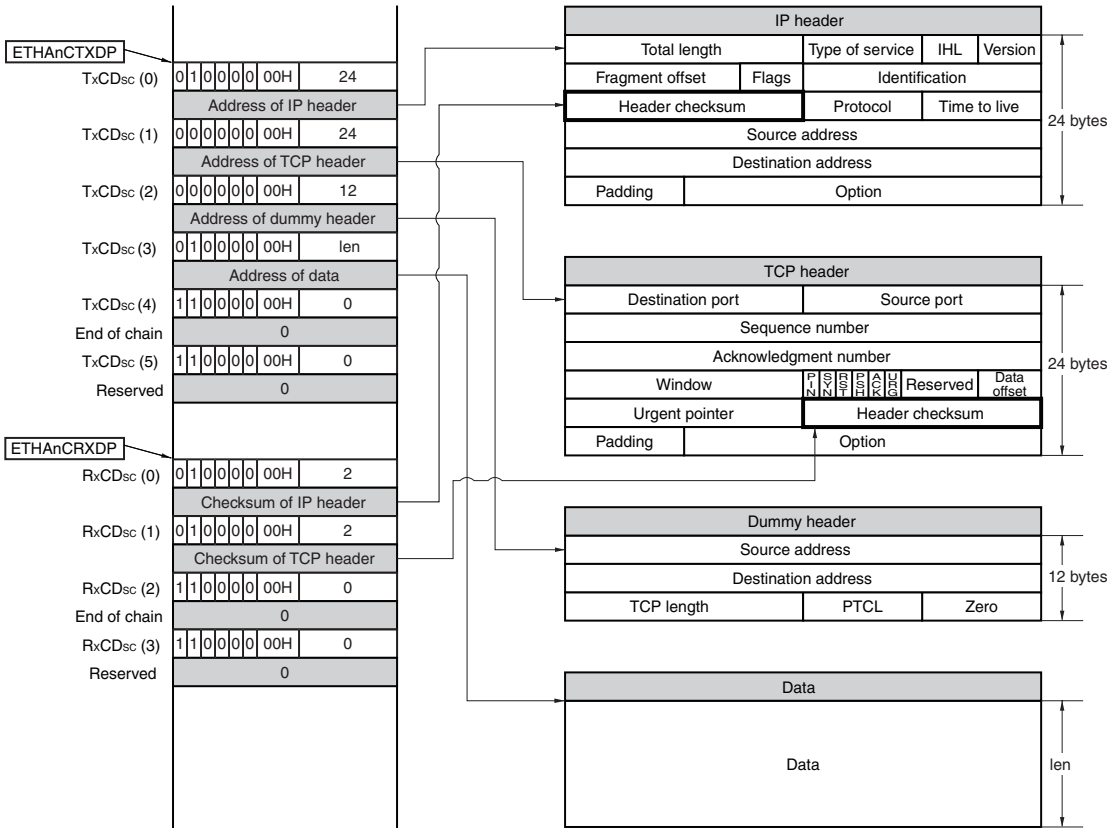
A checksum is calculated and the result is formatted in little endian format.

23.8.1 Configuration of transmit checksum descriptor

Calculation of a transmit checksum is controlled by a descriptor.

Depending on the configuration of the descriptor, the IP header and TCP header can be calculated separately or all at once.

The descriptor configuration for calculating the IP header and TCP header all at once is shown below.



The address to which the checksum of the IP header is to be written is obtained by adding 10 bytes to the IP header address, and the address to which the checksum of the TCP header is to be written is obtained by adding 16 bytes to the TCP header address.

Figure 23-33 Configuration of transmit checksum descriptor

23.9 Cautions

23.9.1 Cautions on FIFO

Note the following restrictions on the internal FIFO buffers of the Ethernet controller.

Table 23-121 Restrictions on transmit FIFO

Maximum FIFO capacity	DMA transfer condition	Retry/abort	Condition to transmit data to PHY	Feature	Note
2,044 bytes or less	Data in transmit FIFO is less than 1,536 bytes[a].	Data is automatically retransmitted/ aborted when a collision is detected.	At least one packet is in the transmit FIFO.	Data can be retransmitted without underrun.	The transmit packet length must be 1,536 bytes or less.

a) Multiple packets may be stored in the transmit FIFO as long as its capacity is not exceeded. However, if the data in the transmit FIFO reaches 1,536 bytes, DMA transmission stops, preventing the transmit FIFO from overflowing. However, because the Ethernet controller starts transmission after one packet of data has been stored in the transmit FIFO, the transmit FIFO is locked if the length of one packet is more than 1,536 bytes. Be sure to observe the rated size of one packet to use (1,518 bytes or less for non-VLAN frames or 1,522 bytes or less for VLAN frames).

Table 23-122 Restrictions on receive FIFO

Maximum FIFO capacity	DMA transfer condition	Condition to transmit pause control frame[a]	Feature	Note
2,036 bytes or less	At least one packet is in the receive FIFO.	• Transmission of pause control frame: Data in the receive FIFO is greater than the size specified by the FLOWTHR bits. • Transmission of 0 pause control frame: Data in the receive FIFO is less than the size specified by the ZPTHR bits.	All error packets can be discarded.	The receive packet length must be 2,036 bytes or less.

a) Control by a pause control frame does not completely prevent the receive FIFO from overflowing. If the receive FIFO overflows, receive packets that are not accepted are discarded.

Chapter 24 Clocked Serial Interface G (CSIG)

This chapter describes clocked serial interface G (CSIG).

The first section describes all V850E2/Sx4-H specific properties, such as instances, register base addresses, andinput/output signal names.
The subsequent sections describe the features that apply to all implementations.

24.1 V850E2/Sx4-H CSIG Features

Instances

This microcontroller has the following number of instances of CSIG:

Table 24-1 Instances of CSIG

CSIG	
Number of instances	2
Name	CSIG0, CSIG4

Instances index n

Throughout this chapter, the individual instances of CSIG is identified by the index "n" (n = 0 or 4), for example, CSIGnCTL0 for CSIGn control register 0.

Register addresses

All CSIGn register addresses are given as addresses offset from the individual base address <CSIGn_base>.
The base address <CSIGn_base> of each CSIGn is listed in the following table:

Table 24-2 Register base addresses <CSIGn_base>

CSIGn	<CSIGn_base> address
CSIG0	FF70 0000_H
CSIG4	FF74 0000_H

Clock supply

The following clock is supplied to CSIG:

Table 24-3 CSIGn clock supply

CSIGn	Clock	Connected to:
CSIG0	PCLK	Clock generator CKSCLK_108
CSIG4	PCLK	Clock generator CKSCLK_011

Maximum transfer speed (baud rate)

For CSIG, communication is possible at the maximum transfer speeds (baud rates) listed in the following table.

Table 24-4 Maximum CSIGn transfer speeds (baud rates)

Mode	Maximum transfer speed (baud rate)
Master mode	10 MHz
Slave mode	8 MHz

Interrupts and DMA

CSIG can generate the following interrupt requests and DMA requests:

Table 24-5 CSIGn interrupt requests and DMA requests

CSIGn signals	Function	Connected to:
CSIG0:		
CSIG0TIC	Communication status interrupt	Interrupt controller INTCSIG0IC DMA controller trigger 43
CSIG0TIR	Reception status interrupt	Interrupt controller INTCSIG0IR DMA controller trigger 42
CSIG0TIRE	Reception error interrupt	Interrupt controller INTCSIG0IRE
CSIG4:		
CSIG4TIC	Communication status interrupt	Interrupt controller INTCSIG40IC DMA controller trigger 83
CSIG4TIR	Reception status interrupt	Interrupt controller INTCSIG4IR DMA controller trigger 82
CSIG4TIRE	Reception error interrupt	Interrupt controller INTCSIG4IRE

CSIG hardware reset

CSIG and its registers are initialized by the following reset signal:

Table 24-6 CSIGn reset signal

CSIGn	Reset signal
CSIGn	System reset SYSRES

I/O signals The I/O signals of CSIG are listed in the following table:

Table 24-7 CSIGn I/O signals

CSIGn signal	Function	Connected to:
CSIG0:		
CSIG0TSCK	Serial clock signal	Port CSIG0SC
CSIG0TSI	Serial data input signal	Port CSIG0SI
CSIG0TSO	Serial data output signal	Port CSIG0SO
CSIG0TSSI	Slave select input signal	Port $\overline{\text{CSIG0SSI}}$
CSIG0TRY	Ready/busy signal	Not connected
CSIG4:		
CSIG4TSCK	Serial clock signal	Port CSIG4SC
CSIG4TSI	Serial data input signal	Port CSIG4SI
CSIG4TSO	Serial data output signal	Port CSIG4SO
CSIG4TSSI	Slave select input signal	Port $\overline{\text{CSIG4SSI}}$
CSIG4TRY	Ready/busy signal	Not connected

<R> **Caution** Port filters are assigned to the input pins of clocked serial interface G (CSIGn), and these filters are enabled as the initial setting. However, because the use of the port filters might cause a communication error, do not use the port filters when using CSIGn; instead, enable the filter bypass by setting the corresponding registers as shown below.

$\overline{\text{CSIG0SC}}$: FCLA24CTL0 = 80H, CSIG0SI: FCLA24CTL2 = 80H, $\overline{\text{CSIG0SSI}}$: FCLA24CTL3 = 80H, $\overline{\text{CSIG4SC}}$: FCLA7CTL2 = 80H, CSIG4SI: FCLA7CTL3 = 80H, $\overline{\text{CSIG4SSI}}$: FCLA7CTL5 = 80H

Data consistency check The following table shows the CSIGnSO ports and their capability to use them for data consistency checks. See *24.3.10 "Error detection"* for details about data consistency checks.

Table 24-8 CSIGn data consistency check ports

CSIGn I/O port signal	Port	Alternative function	Data consistency check
CSIG0:			
CSIG0SO	P0_1	ALT_OUT3	Not possible
	P21_5	ALT_OUT3	Possible
CSIG4:			
CSIG4SO	P0_2	ALT_OUT3	Possible
	P21_13	ALT_OUT3	Possible

24.2 Functional Overview

Features summary

- Three-wire serial synchronous data transfer
- Master mode and slave mode selectable
- Slave select input signal ($\overline{\text{CSIGnTSSI}}$)
- Built-in baud rate generator
- Adjustable baud rate; in slave mode it is determined by the input clock
- Maximum transmission speed:
  - in master mode: PCLK/4
  - in slave mode: PCLK/6

Caution There might be restrictions on the maximum baud rate that can actually be used depending on the product. Specify the baud rate so as not to exceed the maximum rate for each product.

- Phase of clock and data selectable
- Data transfer with MSB or LSB first selectable
- Transfer data length selectable from 7 to 16 bits in 1-bit units
- Extended data length (EDL) function for transferring more than 16 bits of data
- Three selectable transfer modes:
  - Transmission mode
  - Reception mode
  - Transmission/reception mode
- Built-in handshake function
- Separate transmit and receive buffers (two 16-bit registers)
- Error detection (data consistency check, parity, overrun)
- Three different interrupt request signals (CSIGnTIC, CSIGnTIR, CSIGnTIRE)
- Loop back mode (LBM) function for self test

The block diagram shows the main components of the CSIG.

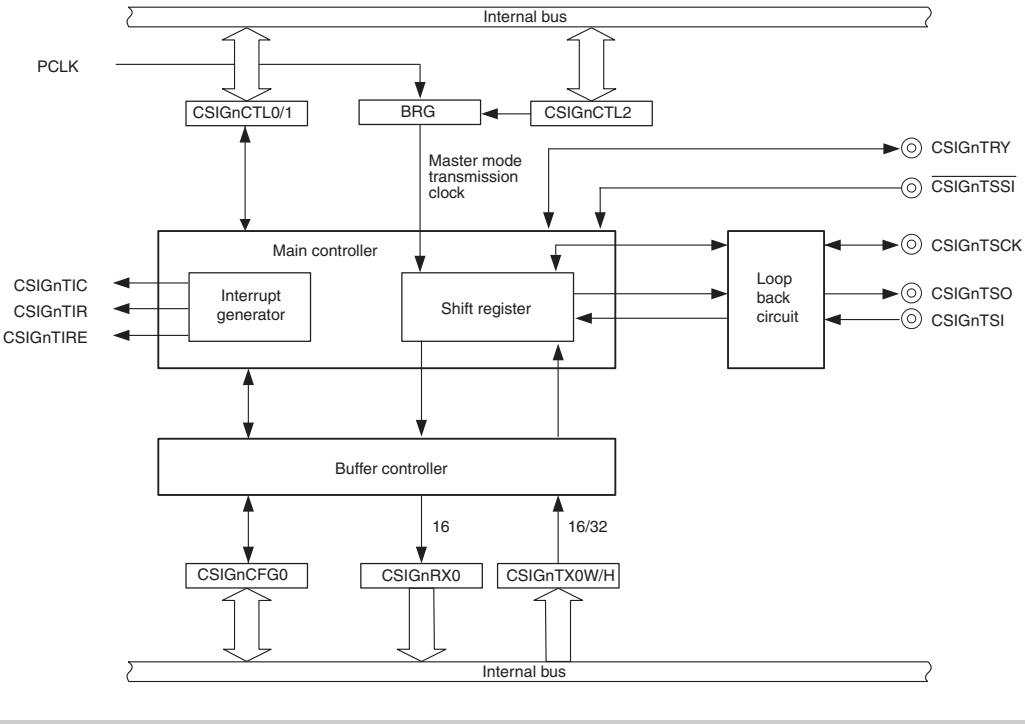


Figure 24-1 CSIG block diagram

In master mode, the serial clock CSIGnTSCK is generated by the built-in baud rate generator (BRG). In slave mode, the serial clock is supplied from an external source.

24.3 Functional Description

The Clocked Serial Interface G uses three signals for communication:

- Serial clock CSIGnTSCK (output for the master mode or input for the slave mode)
- Data output signal CSIGnTSO
- Data input signal CSIGnTSI

Additional signals are available for external control and monitoring.

- $\overline{\text{CSIGnTSSI}}$: Slave select input signal
- CSIGnTRY: Handshake signal (input for the master mode or output for the slave mode)

Whether the CSIG operates in the master or slave mode can be specified by using the CSIGCTL2 register.

Data transmission is bit-wise and serial and synchronous to the serial clock.

The most important registers for setting up the CSIG are:

Register	Function
CSIGnCTL0	Enables/disables the operation clock (PCLK), data transmission, and data reception
CSIGnCTL1	Controls options like interrupt timing, extended data length, data consistency check, loop-back mode, handshake, etc.
CSIGnCTL2	Selects master/slave mode and – effective in master mode – the baud rate of the internal baud rate generator (BRG)
CSIGnCFG0	Configures the communication protocol

24.3.1 Operating modes (master/slave)

Master/slave selection is performed by using the CSIGnCTL2.CSIGnPRS[2:0] bits, and, when the master is selected, the source clock of the serial clock must also be selected.

(1) Master mode

In the master mode, the serial clock is generated by the internal baud rate generator (BRG) and provided to the slave(s) by signal CSIGnTSCK.

Master mode is enabled by setting CSIGnCTL2.CSIGnPRS[2:0] to anything but 111_B. In the master mode, the BRG frequency can be specified by specifying values for the CSIGnCTL2.CSIGnPRS[2:0] and CSIGnCTL2.CSIGnBRS[11:0] bits in combination.

Clock defaults The default level of CSIGnTSCK depends on the clock phase selection bit: It is high when CSIGnCTL1.CSIGnCKR = 0, and is low when CSIGnCTL1.CSIGnCKR = 1.

The example below shows the communication in master mode for eight data bits when CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0, and MSB first:

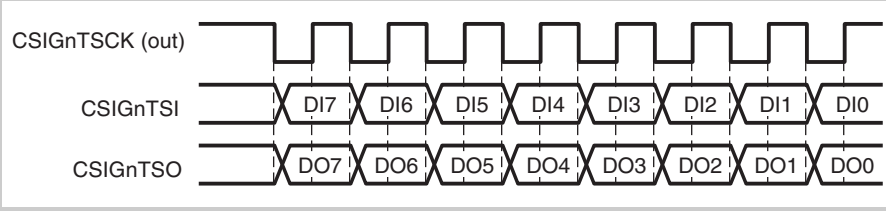


Figure 24-2 Transmission/reception in master mode

(2) Slave mode

In the slave mode, another device is the communication master. The external clock is supplied through the CSIGnTSCK signal. When the serial clock signal is detected, a transmission or reception operation immediately starts.

Slave mode is selected by setting CSIGnCTL2.CSIGnPRS[2:0] to 111_B.

Note When using the slave mode, the baud rate generator (BRG) can be disabled by clearing the CSIGnCTL2.CSIGnBRS[11:0] bits, reducing power consumption.

The example below shows the communication in the save mode for eight data bits when CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0, and the MSB is first.

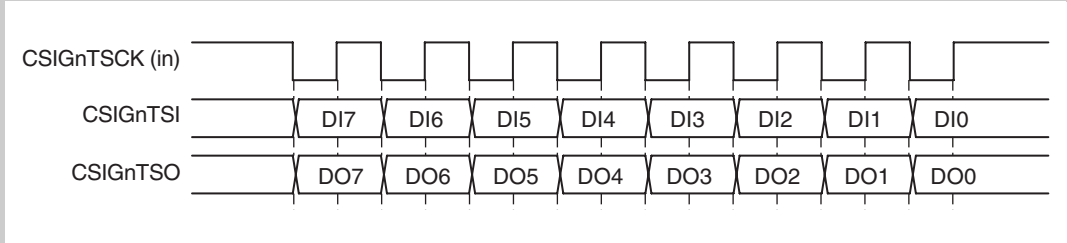


Figure 24-3 Transmission/reception in slave mode

24.3.2 Master/slave connections

(1) One master and one slave

The following figure illustrates the connections between one master and one slave.

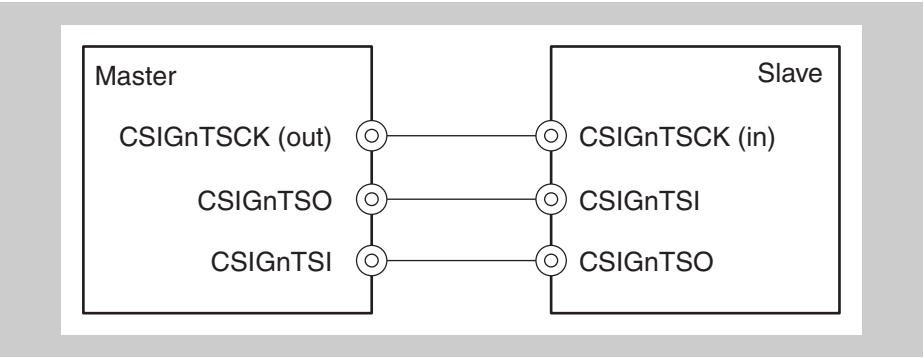


Figure 24-4 Simple master/slave connection

(2) One master and multiple slaves

The following figure illustrates the connections between one master and multiple slaves. In this example, a configuration in which the master supplies one slave select (SS) signal to each slave is possible. This signal is connected to the slave select input $\overline{\text{CSIGnTSSI}}$ of the slave.

The $\overline{\text{CSIGnTSSI}}$ signal recognition function can be enabled/disabled by bit CSIGnCTL1.CSIGnSSE.

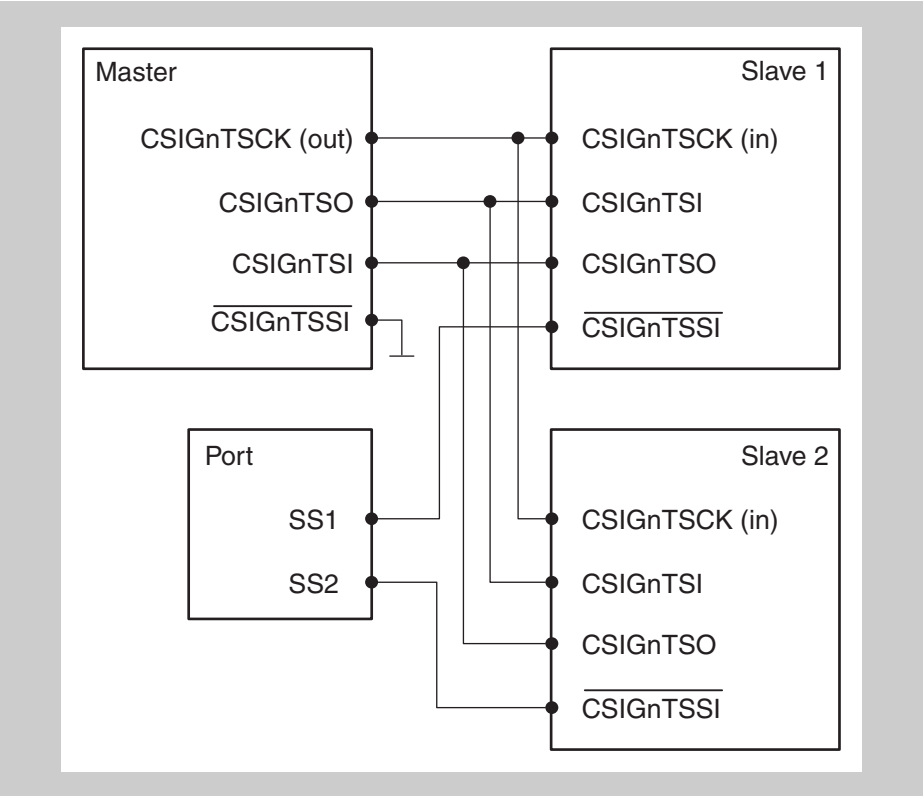


Figure 24-5 Master to multiple slaves connection

A slave is selected (enabled) when its $\overline{\text{CSIGnTSSI}}$ signal is low.

If a slave is not selected, it will neither receive nor transmit data. In addition, its output CSIGnTSO is set to input mode in order to avoid interference with the output of another slave that was selected.

(3) CSIGnTSO output control

CSIG can output CSIGnTSO when all of the following conditions are satisfied:

- The CSIG is enabled (CSIGnCTL0.CSIGnPWR = 1).
- The CSIG is operated in transmission or transmission/reception mode (CSIGnCTL0.CSIGnTXE = 1).
- The CSIG is operated with slave select enabled (CSIGnCTL1.CSIGnSSE = 1).
- The slave mode selection signal $\overline{\text{CSIGnTSSI}}$ is inactive, i.e. on high level.

By this signal congestions on the external CSIGnTSO signal line are avoided.

24.3.3 Serial clock selection

In master mode, the transmission baud rate is selectable using the CSIGnPRS[2:0] and CSIGnBRS[11:0] bits in the CSIGnCTL2 register.

The following figure shows a block diagram of the BRG.

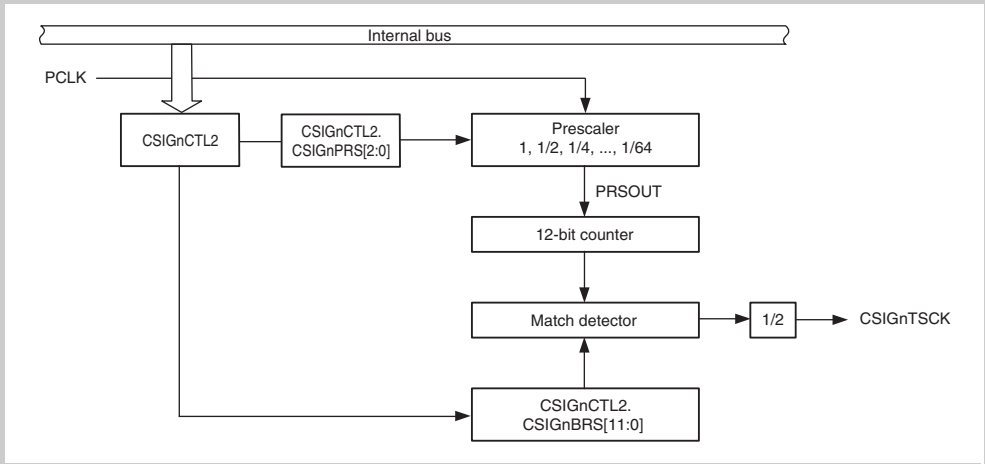


Figure 24-6 BRG block diagram

Clearing CSIGnCTL2.CSIGnBRS[11:0] disables the BRG.

Baud rate calculation

The baud rate is calculated as: PCLK / (2^m × k × 2), where

m = CSIGnPRS[2:0] = 0 to 6

k = CSIGnBRS[11:0] = 1 to 4095

Baud rate limits

When setting the baud rate, please note:

- Maximum acceptable baud rate in master mode is PCLK / 4.
- Maximum acceptable baud rate in slave mode is PCLK / 6.
- Minimum baud rate in both modes is PCLK / 524160.

Caution There might be restrictions on the maximum baud rate that can actually be used depending on the product. Specify the baud rate so as not to exceed the maximum rate for each product.

Example

If PCLK = 80 MHz, the maximum baud rate is

- 20.0 Mbps (PCLK / 4) in master mode
- 13.3 Mbps (PCLK / 6) in slave mode

The slowest baud rate is 152.625 bps (PCLK / 524160).

24.3.4 Data transfer modes

(1) Transmission mode

Setting CSIGnCTL0.CSIGnTXE = 1 and CSIGnCTL0.CSIGnRXE = 0 puts the CSIG in transmission mode. Transmission starts when transmission data is written in the CSIGnTX0W or CSIGnTX0H register.

(2) Reception Mode

Setting CSIGnCTL0.CSIGnTXE = 0 and CSIGnCTL0.CSIGnRXE = 1 puts the CSIG in reception mode.

In the master mode, reception starts when dummy data is read from the CSIGnRX0 register.
All following receptions are triggered by a read from the reception data register CSIGnRX0, as long as CSIGnBCTL0.CSIGnSCE = 1.

In the slave mode, reception starts when the serial clock CSIGnTSCK is supplied from the master.

Note In reception mode, any previously received data must be read from the reception data register CSIGnRX0 in order to avoid any overwrite situation.

Moreover the communication start bit CSIGnBCTL0.CSIGnSCE has to be set to 1 and has to set back to 0 before reading the last received data from CSIGnRX0.

The recommended procedure is:

1. Set CSIGnBCTL0.CSIGnSCE = 1.
2. Read CSIGnRX0 (dummy data).
3. Wait for the reception interrupt CSIGnTIR.
4. Read CSIGnRX0 (received data).
   In case of further data receptions continue at step 3, repeat a read operation until all data has been received.
   Before reading the last received data from CSIGnRX0, clear CSIGnBCTL0.CSIGnSCE.

(3) Transmission/reception mode

Setting CSIGnCTL0.CSIGnTXE = 1 and CSIGnCTL0.CSIGnRXE = 1 puts the CSIG in transmission/reception mode.

Communication starts when transmission data is written to the CSIGnTX0W or CSIGnTX0H register.

24.3.5 Data length selection

(1) Data length between 7 and 16 bits

Transmission data length is selectable from 7 to 16 bits using the CSIGnCFG0.CSIGnDLS[3:0] bits. The examples below show the communication with MSB first (CSIGnCFG0.CSIGnDIR = 0):

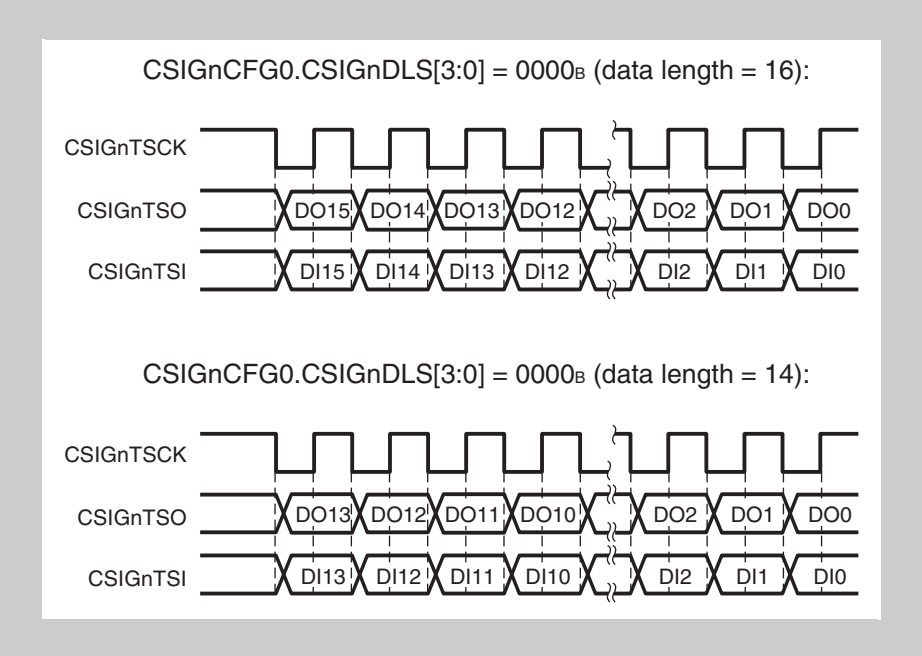


Figure 24-7 Data length select function

(2) Data length greater than 16 bits

If the length of the data to be sent/received exceeds 16 bits, the extended data length (EDL) feature can be used.

The EDL function is enabled by setting CSIGnCTL1.CSIGnEDLE.

The EDL function works as follows:

- Data is divided into 16-bit blocks and a remainder. For example, a 42-bit character string is divided into two 16-bit blocks and a 10-bit remainder.
- For the remainder, the data length is specified for the CSIGnCFG0.CSIGnDLS[3:0] bits.
- When transmitting 16-bit blocks, set the CSIGnTX0W.CSIGnEDL bit. In this case, the data written to CSIGnTX0W is sent as a 16-bit data length regardless of the CSIGnCFG0.CSIGnDLS[3:0] bits.
- When the specified length of data (the remainder when CSIGnTX0W.CSIGnEDL = 0) is transmitted, the transfer ends.

Example Example of transmitting the 40-bit data $123456789A_H$

The 40-bit data is divided into two 16-bit blocks of data and one 8-bit block of data.

- Initialize CSIGnCFG0.CSIGnDLS[3:0] = 8_D.
- To send the string $123456789A_H$ with MSB first, write the following sequence to CSIGnTX0W:
  - $2000\ 1234_H$ (CSIGnTX0W.CSIGnEDL = 1)
  - $2000\ 5678_H$ (CSIGnTX0W.CSIGnEDL = 1)
  - $0000\ 009A_H$ (CSIGnTX0W.CSIGnEDL = 0)

The following figure illustrates the timing.

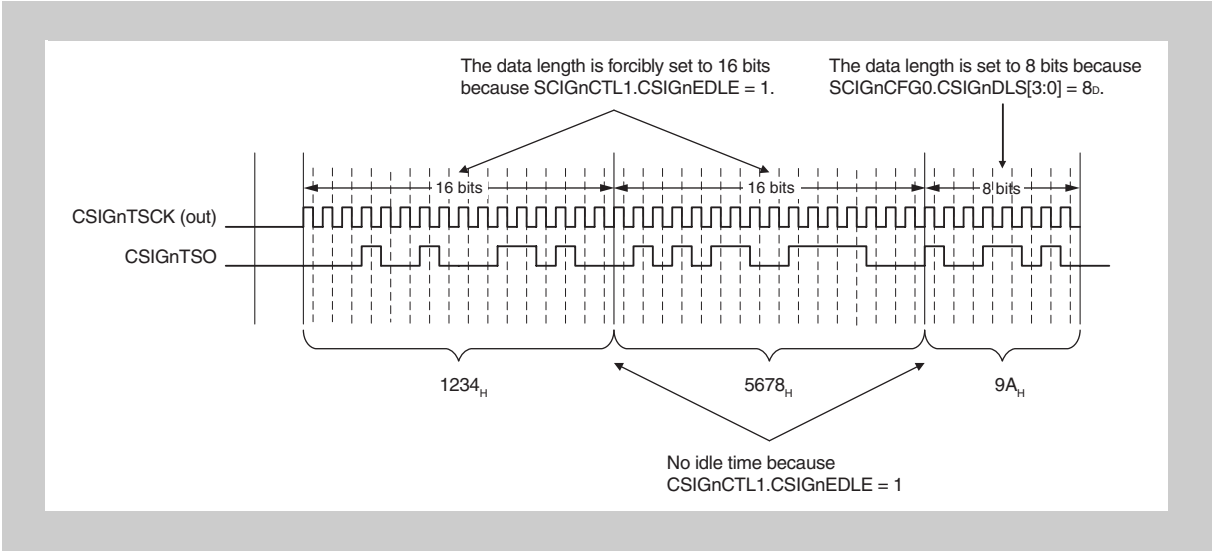


Figure 24-8 EDL timing chart

Notes

1. A data length less than 7 bits can be specified only when the EDL function is used.
2. It is not possible to send two consecutive data with a data length of less than 7 bits.
3. If parity is enabled, the parity bit is added after the last bit.
4. The following describes an example where the transmitted data is 123456_H.
   - CSIGnCFG0.CSIGnDIR is cleared to 0 (MSB first).
     $2000\ 1234_H$ is written to CSIGnTX0W (CSIGnTX0W.CSIGnEDL = 1).
     $0000\ 0056_H$ is written to CSIGnTX0W (CSIGnTX0W.CSIGnEDL = 0).
   - CSIGnCFG0.CSIGnDIR is set to 1 (LSB first).
     $2000\ 3456_H$ is written to CSIGnTX0W (CSIGnTX0W.CSIGnEDL = 1).
     $0000\ 0012_H$ is written to CSIGnTX0W (CSIGnTX0W.CSIGnEDL = 0).
5. The EDL function cannot be used in the slave mode (CSIGnCTL1.CSIGnPRS[2:0] = 1, 1, 1) and reception mode (CSIGnCTL0.CSIGnTXE = 0, CSIGnCTL0.CSIGnRXE = 1).

24.3.6 Serial data direction selection

The serial data direction is selectable using the CSIGnDIR bit in the CSIGnCFG0 register.

The examples below show the communication for a data length of 8 bit (CSIGnCFG0.CSIGnDLS[3:0] = 1000_B):

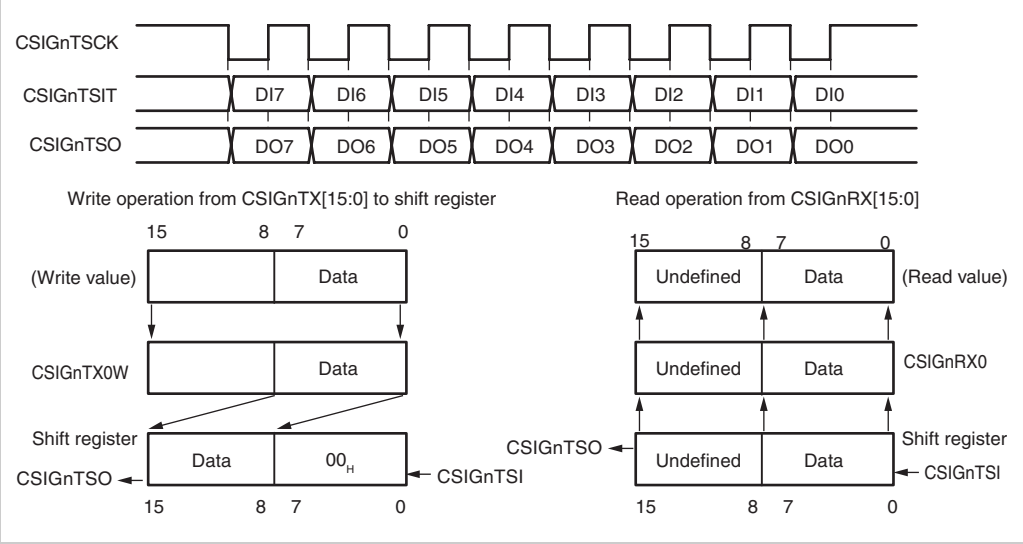


Figure 24-9 Serial data direction select function - MSB first (CSIGnDIR = 0)

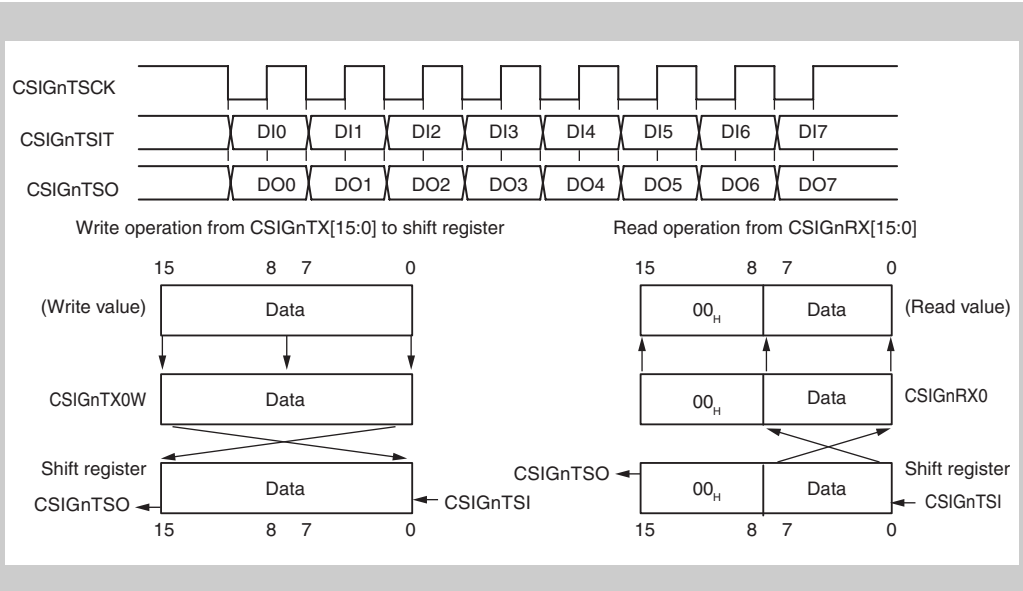


Figure 24-10 Serial data direction select function - LSB first (CSIGnDIR = 1)

24.3.7 Communication in slave mode

The following figure illustrates the communication signals and timings in slave mode.

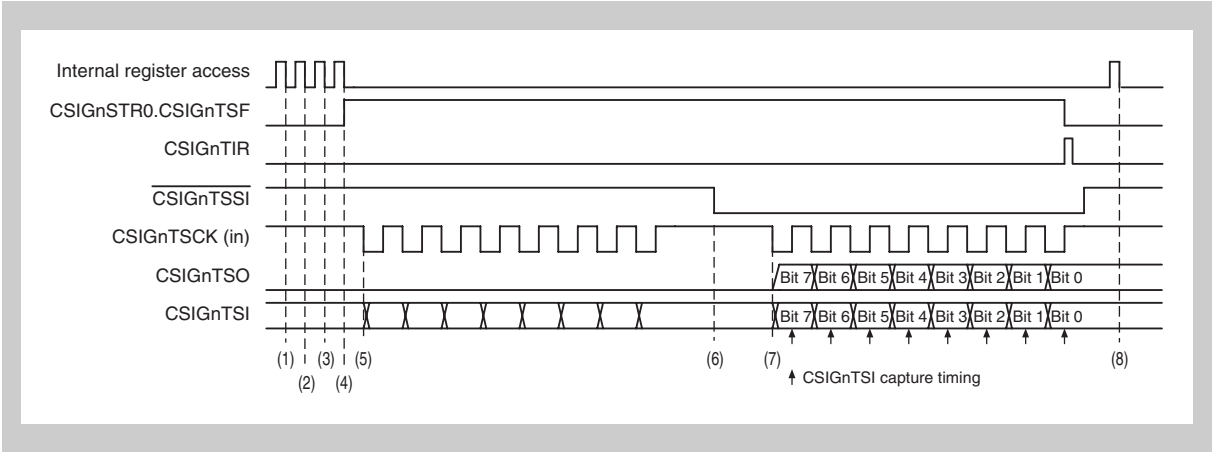


Figure 24-11 Transmit/receive communication timing in slave mode

1. CSIG is placed in the slave mode by setting CSIGnCTL2.CSIGnPRS[2:0] to 111_B. The $\overline{\text{CSIGnTSSI}}$ signal is enabled (CSIGnCTL1.CSIGnSSE = 1).
2. CSIGnCTL1.CSIGnCKR and CSIGnCFG0.CSIGnDAP0 are 0, the data length is 8 bits (CSIGnCFG0.CSIGnDLS0[3:0] = 1000_B), and the data direction is MSB first (CSIGnCFG0.CSIGnDIR0 = 0).
3. CSIG is set to the transmission/reception mode (CSIGnCTL0.CSIGnPWR = 1, CSIGnCTL0.CSIGnTXE = 1, and CSIGnCTL0.CSIGnRXE = 1). Communication is enabled to start.
4. If transfer data is written to the transmission data register CSIGnTX0W or CSIGnTX0H, the transfer status flag CSIGnSTR0.CSIGnTSF is automatically set, and the system waits for the $\overline{\text{CSIGnTSSI}}$ signal to become low.
5. Data transmission/reception does not start while the $\overline{\text{CSIGnTSSI}}$ signal is high, even if the external serial clock CSIGnTSCK is input. CSIGnTSO retains the value, and input to CSIGnTSI is ignored.
6. When the $\overline{\text{CSIGnTSSI}}$ signal becomes low, it indicates that CSIGnTSO is enabled and data transmission is possible.
7. If a serial clock is input while $\overline{\text{CSIGnTSSI}}$ is low, the transfer data is transmitted from CSIGnTSO in synchronization with the serial clock, and data is received from CSIGnTSI at the same time.
8. The CSIGnRX0 register is read.

24.3.8 CSIG interrupts

CSIG can generate the following interrupts:

- CSIGnTIC (communication interrupt)
- CSIGnTIR (reception interrupt)
- CSIGnTIRE (error interrupt)

(1) CSIGnTIC (communication interrupt)

This interrupt is normally generated after every data transfer. It can be used to trigger a DMA for writing new transmission data to register CSIGnTX0W or CSIGnTX0H.

The following example assumes:

- Master mode
- CSIGnCTL1.CSIGnSIT = 0 (no interrupt delay)
- CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0 (normal clock and data phase)
- CSIGnCFG0.CSIGnDLS[3:0] = 1000_B (data length 8 bits)
- CSIGnCTL1.CSIGnSLIT = 0 (normal interrupt timing)

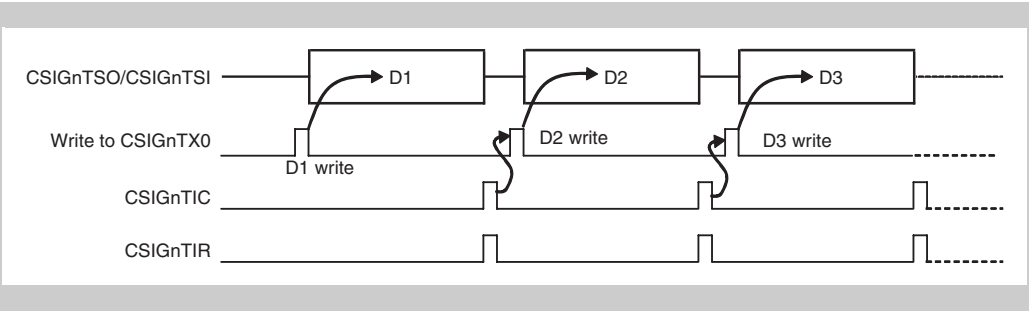


Figure 24-12 Generation of CSIGnTIC after communication (CSIGnCTL1.CSIGnSLIT = 0)

However, CSIGnTIC can also be set up to occur when the CSIGnTX0W or CSIGnTX0H register is free for the next data. This is specified by setting CSIGnCTL1.CSIGnSLIT = 1.

This mode allows more efficient data transfers.

The effect is illustrated in the figure below.

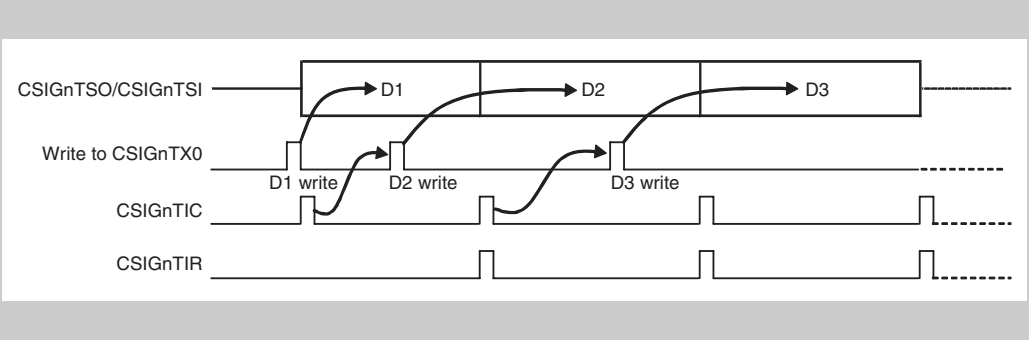


Figure 24-13 Generation of CSIGnTIC at the beginning of communication

(2) CSIGnTIR (reception interrupt)

This interrupt is generated in reception and transmission/reception mode after data has been received and is available in the reception data register.

The following example assumes:

- Master mode
- CSIGnCTL1.CSIGnSIT = 0 (no interrupt delay)
- CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0 (normal clock and data phase)
- CSIGnCFG0.CSIGnDLS[3:0] = 1000_B (data length 8 bits).

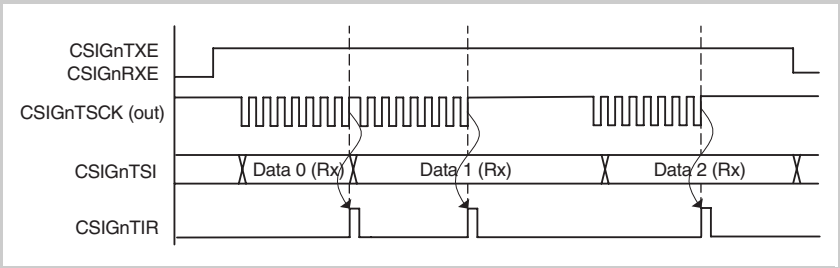


Figure 24-14 Generation of CSIGnTIR

(3) CSIGnTIRE (reception error interrupt)

This interrupt is generated whenever an error is detected.

Table 24-9 Data error types

Error type	Communication status after error interrupt
Parity error	Interrupt is generated and communication continues
Data consistency error	Interrupt is generated and communication continues
Overrun error	If CSIGnCTL1.CSIGnHSE = 0 (no handshake), communication continues after the interrupt is generated. (Communication does not stop.)
	If CSIGnCTL1.CSIGnHSE = 1 (handshake exists), communication is stopped according to the handshake. No interrupt is generated, and no overrun error occurs.

The type of error that caused the generation of CSIGnTIRE is indicated in register CSIGnSTR0.

For details about the various error types, see *24.3.10 “Error detection” on page 1808*.

(4) Interrupt delay

In the master mode, all interrupts generated by the master can be delayed one half cycle of the serial clock CSIGnTSCK. This function cannot be used in the slave mode.

To specify this delay, set the CSIGnCTL1.CSIGnSIT bit to 1.

The figure below shows an example of using the interrupt delay function with the following settings: CSIGnCTL1.CSIGnSIT = 1 (interrupt delay enabled), CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0 (normal clock phase and data phase), and CSIGnCFG0.CSIGnDLS[3:0] = 1000_B (8-bit data length).

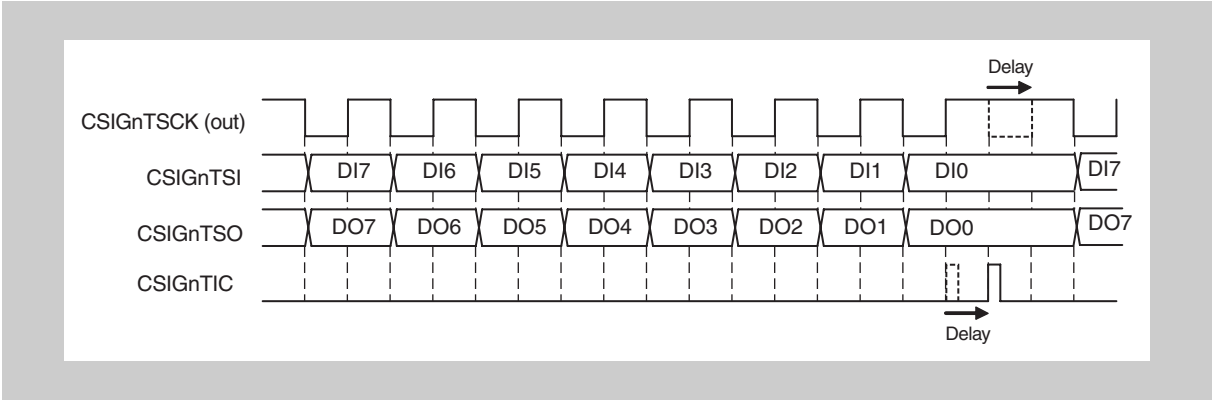


Figure 24-15 Interrupt delay function (CSIGnCTL1.CSIGnSIT = 1)

24.3.9 Handshake function

CSIG features a handshake function to synchronize the master and the slave devices. This function can be enabled/disabled by bit CSIGnCTL1.CSIGnHSE. For handshake, the signal CSIGnTRY is used. In addition, when the handshake function is disabled (CSIGnCTL1.CSIGnHSE = 0), the CSIGnTRY signal is output at the low level.

The timing depends on the data phase selection bit CSIGnCFG0.CSIGnDAP.

(1) Slave mode

When CSIGnCTL1.CSIGnHSE = 1 and the slave is busy, the CSIGnTRY signal is output at the low level. This happens when previous receive data is still in the CSIGnRX0 register, and new data cannot be copied from the shift register to CSIGnRX0 (CSIGnRX0 full condition).

The following examples assume a data length of 8 bits.

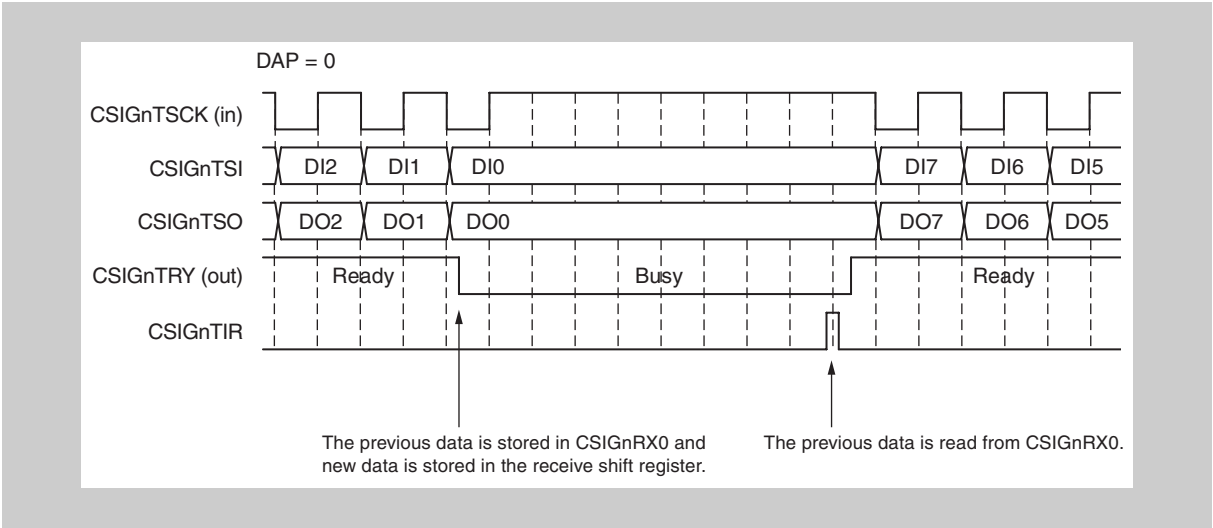


Figure 24-16 Ready/busy signal from slave (CSIGnCFG0.CSIGnDAP = 0)

As long as the slave is busy, the master has to wait (i.e. suspend the serial clock). The slave sets CSIGnTRY to high (“ready”) as soon as the reception data register CSIGnRX0 has been read.

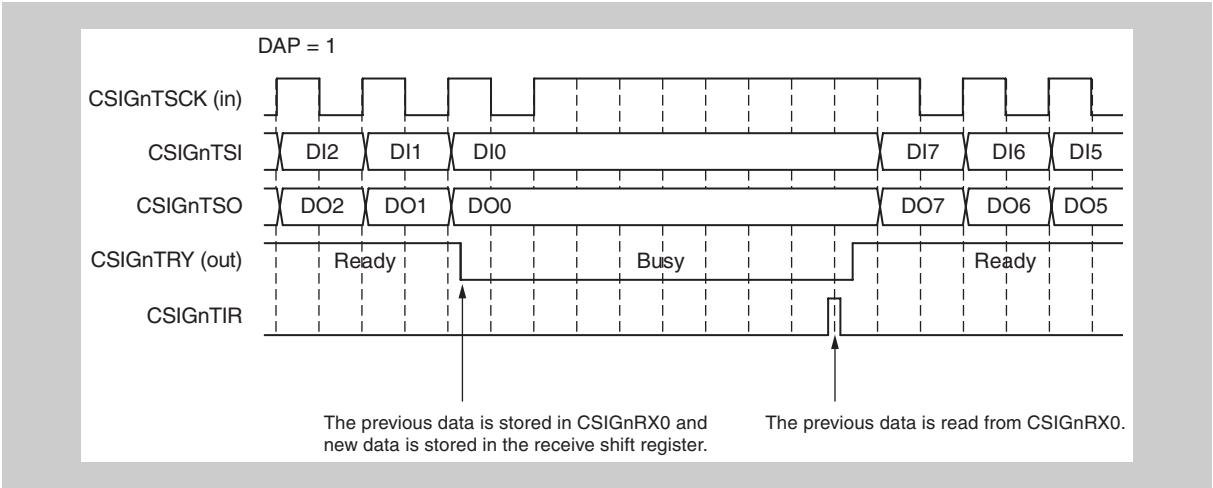


Figure 24-17 Ready/busy signal from slave (CSIGnCFG0.CSIGnDAP = 1)

(2) Master mode

When the master detects the CSIGnTRY is at the low level, the following transfer is put on hold, and the master goes into wait status. It suspends the clock CSIGnTSCK.

The CSIGnTRY level is checked at each half clock cycle of CSIGnTSCK.

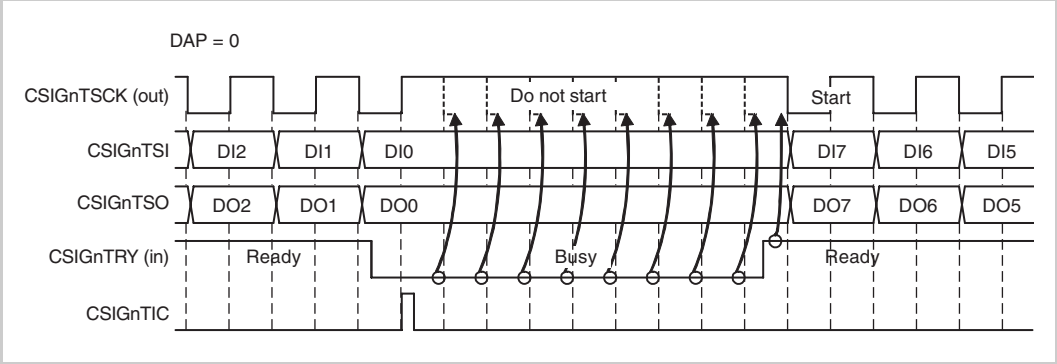


Figure 24-18 Master's reaction to CSIGnTRY (CSIGnCFG0.CSIGnDAP = 0)

If the CSIGnTRY low signal comes from the slave while data transfer is in progress, the serial clock is suspended after the transfer is complete.

The master resumes the communication as soon as CSIGnTRY becomes high (slave is "ready").

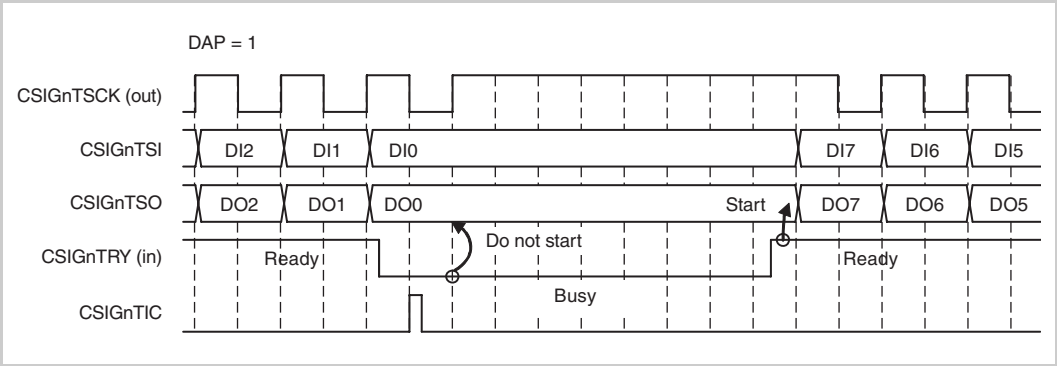


Figure 24-19 Master's reaction to CSIGnTRY (CSIGnCFG0.CSIGnDAP = 1)

Caution If multiple slaves are connected, the master must only detect the CSIGnTRY signal of the slave it has selected for communication.

CSIGnTRY must be pulled down by the external slave before the next transfer starts. Even if the signal is pulled down by the slave during the transfer, the transfer continues.

24.3.10 Error detection

CSIG can detect three error types:

- Data consistency error (transmission data)
- Parity error (reception data)
- Overrun error

Check for data consistency and parity errors can be enabled/disabled individually.

If one of these errors is detected, the interrupt CSIGnTIRE is generated and the corresponding flag is set.

(1) Data consistency check

The purpose of the data consistency check is to ensure that the data physically sent to the output signal is identical with the original data that was copied to the shift register.

The data consistency check can be enabled/disabled by bit CSIGnCTL1.CSIGnDCS. It is not active if data transmission is disabled (CSIGnCTL0.CSIGnTXE = 0).

When the data consistency check is active, the data transferred from CSIGnTX0W or CSIGnTX0H to the shift register is copied to a separate register. In addition, the physical levels at output signal CSIGnTSO are captured, and their logical interpretation is fed into an own shift register.

After completion of the transmission, the data sent is compared with the original transmission data.

Mismatch is considered as a data consistency error.

When a data consistency error occurs:

- Interrupt CSIGnTIRE is generated.
- Bit CSIGnSTR0.CSIGnDCE is set.

The function is illustrated in the following block diagram.

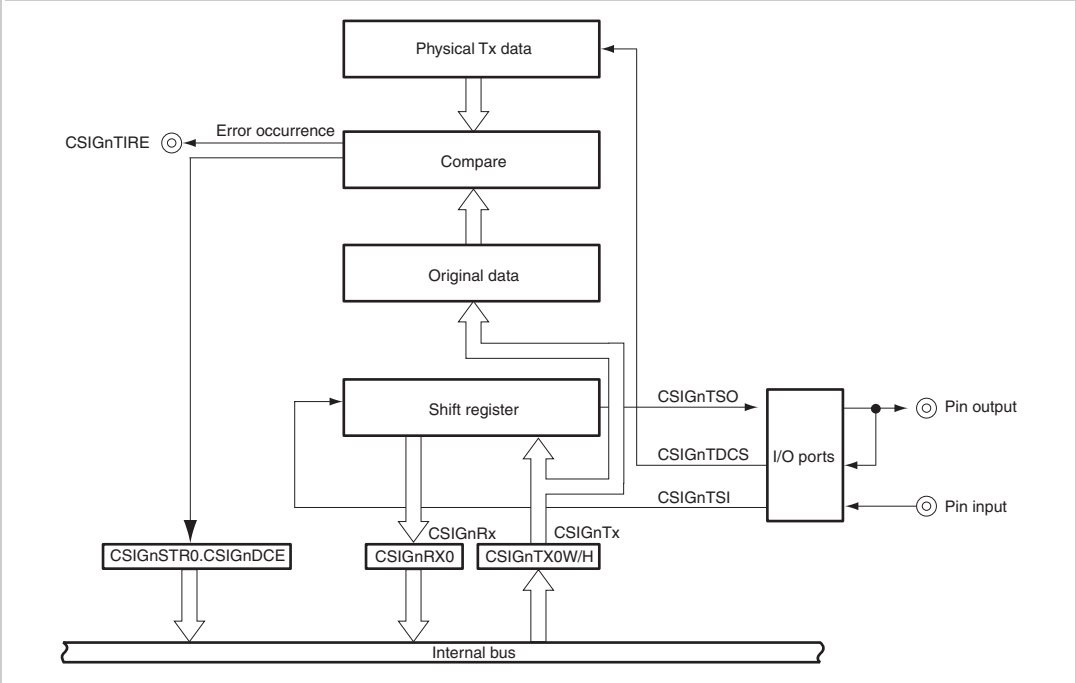


Figure 24-20 Functional block diagram of the data consistency check

(2) Parity check

Parity is a common mean to detect a single bit failure during data transmission. CSIG can append a parity bit to the last data bit (even if extended data length is used).

The use and type of parity is specified in CSIGnCFG0.CSIGnPS[1:0].

Parity check is enabled if CSIGnCFG0.CSIGnPS[1] = 1.

The parity bit is checked after reception is complete.

When a parity error occurs:

- Interrupt CSIGnTIRE is generated
- Bit CSIGnSTR0.CSIGnPE is set

The following figure shows an example.

- Data length is 8 bits.
- The data transmitted is 05_H and 35_H.
- Data direction is LSB first.
- Parity type is odd.

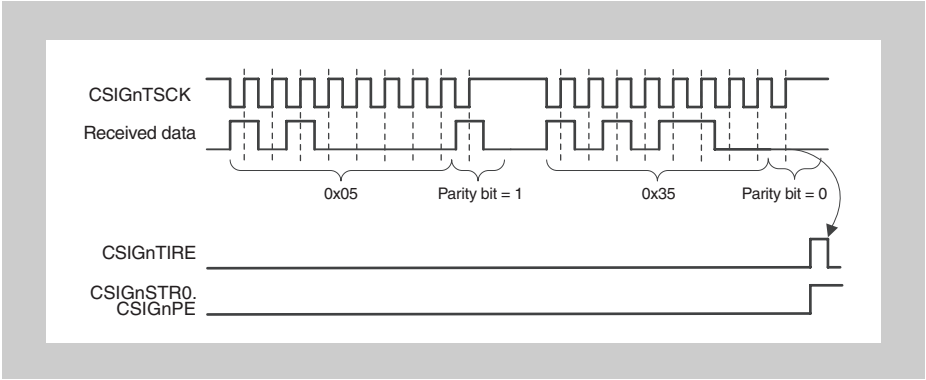


Figure 24-21 Parity check example

For the first 8 bits, the parity bit is 1. There is no parity error, because the total number of ones (including the parity bit) is odd.

For the second 8 bits, the parity bit is 0. This is detected as a parity error, because the total number of ones (including the parity bit) is even.

If using the extended data length (EDL) function, the parity bit is added after the last data bit.

(3) Overrun error

This error occurs when newly received data cannot be transferred from the shift register to the reception data register CSIGnRX0. This happens when CSIGnRX0 was not read and therefore contains previously received data.

In the master mode, because the serial clock is stopped until the CPU reads reception data, overrun errors do not occur. In the slave mode, the handshake function can be used to avoid this error.

When an overrun error occurs:

- The interrupt CSIGnTIRE is generated.
- CSIGnSTR0.CSIGnOVE is set.
- The CSIGnRX0 register is written with the reception data.
- Communication continues.

The following figure illustrates the function.

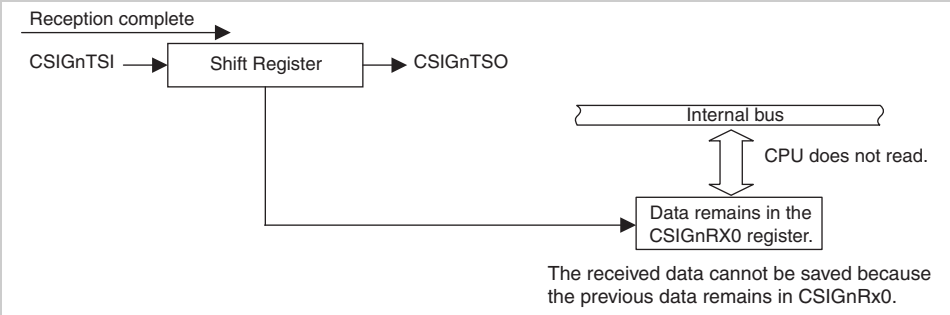


Figure 24-22 Overrun error detection

The following figure illustrates an example where:

- Rx data 3 was not read
- Rx data 4 was received, but cannot be stored

Thus an overrun error occurs.

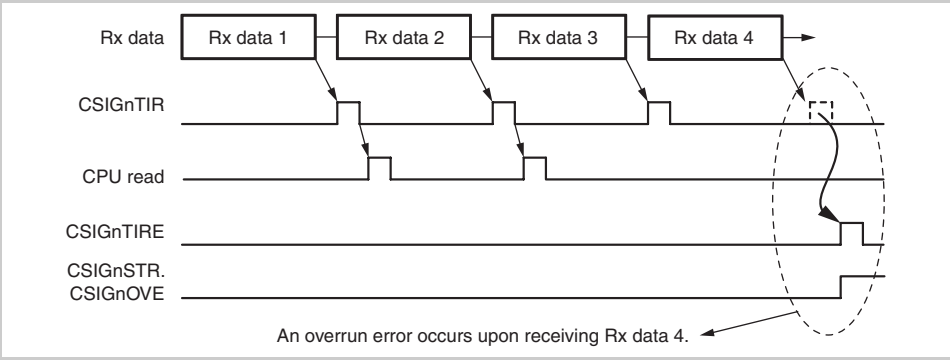


Figure 24-23 Overrun error detection - example

Note An overrun error can be avoided in slave mode by using the handshake.

When handshake is used in slave mode, the receiver (slave) signals to the transmitter (master) that it is busy. The transmitter then waits until the receiver has read its reception data register and is ready again.

For details, see *24.3.9 “Handshake function” on page 1806*.

24.3.11 Loop-back mode

Loop-back mode is a special mode for self-test. This feature is only available in master mode.

When this mode is active, the transmission and reception signals are internally connected, as shown in the figures below. The signals CSIGnTSCK, CSIGnTSO, and CSIGnTSI are disconnected from the ports. In addition, the CSIGnTSO output level is fixed to low, and CSIGnTSCK becomes inactive. The handshake function cannot be used at this time. The rest of CSIG works as in normal operation.

To perform a self-test of the CSIG, CSIGnCTL1.CSIGnLBM is set to 1, and a normal transfer operation is executed. Next, whether the reception data and transmission data are the same is checked.

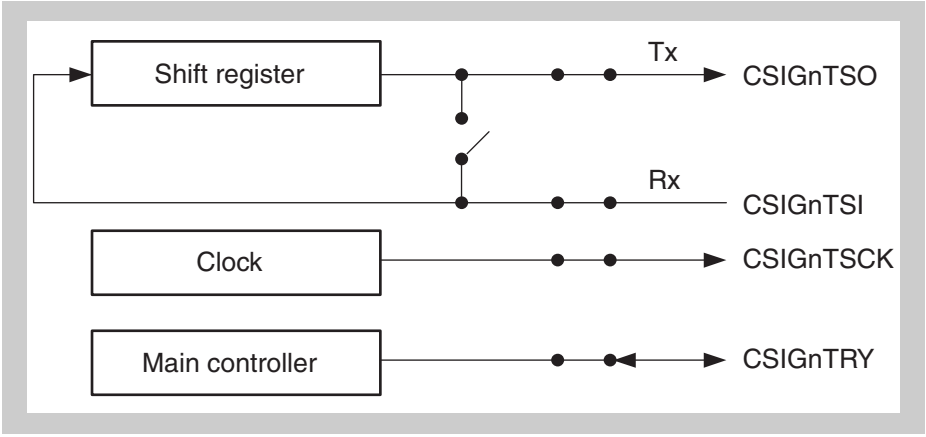


Figure 24-24 Normal operation (CSIGnLBM = 0)

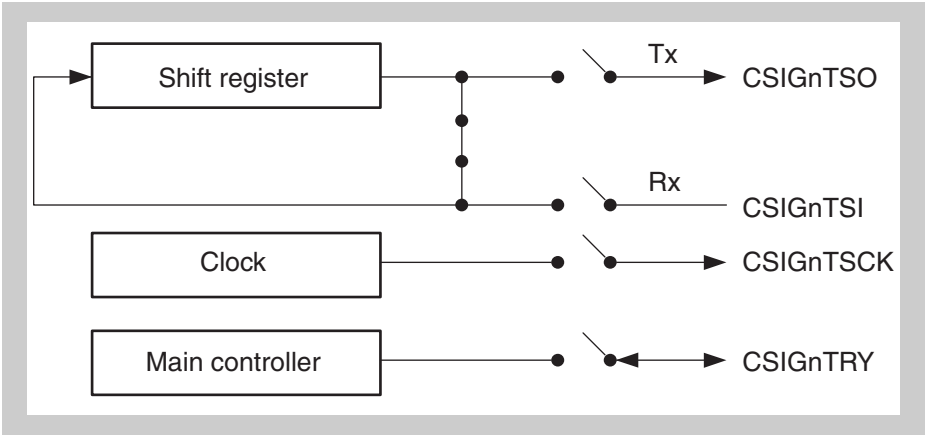


Figure 24-25 Operation in loop-back mode (CSIGnLBM = 1)

24.4 CSIG Control Registers

The CSIGn is controlled and operated by the following registers:

Table 24-10 CSIGn register overview

Register name	Symbol	Address
Control register 0	CSIGnCTL0	<CSIGn_base> + 0000_H
Control register 1	CSIGnCTL1	<CSIGn_base> + 0010_H
Control register 2	CSIGnCTL2	<CSIGn_base> + 0014_H
Status register 0	CSIGnSTR0	<CSIGn_base> + 0004_H
Status clear register 0	CSIGnSTCR0	<CSIGn_base> + 0008_H
Reception mode control register 0	CSIGnBCTL0	<CSIGn_base> + 1000_H
Configuration register 0	CSIGnCFG0	<CSIGn_base> + 1010_H
Transmission data register 0 for word access	CSIGnTX0W	<CSIGn_base> + 1004_H
Transmission data register 0 for half word access	CSIGnTX0H	<CSIGn_base> + 1008_H
<R> Reception data register 0	CSIGnRX0	<CSIGn_base> + $100C_H$

<CSIGn_base> The base addresses <CSIGn_base> of the CSIGn is defined in the first section of this chapter under the key word “Register addresses”.

(1) CSIGnCTL0 - CSIGn control register 0

This register controls the operation clock and enables/disables transmission/ reception.

Access This register can be read/written in 8-bit and 1-bit units.

Address <CSIGn_base> + 0000_H

Initial Value 00_H. This register is initialized by any reset.

Caution Be sure to set “0” to bits 4 to 1 and “1” to bit 0.

7	6	5	4	3	2	1	0
CSIGn PWR	CSIGn TXE	CSIGn RXE	0	0	0	0	a
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

a) Be sure to set “1” to bit 0 though the initial value is 0.

Table 24-11 CSIGnCTL0 register contents

Bit position	Bit name	Function
7	CSIGnPWR	Controls operation clock. 0: Stop operation clock 1: Provide operation clock Clearing CSIGnPWR to 0 resets the internal circuits, stops operation, and sets the CSIG to standby state. No clock is provided to internal circuits. If CSIGnPWR is cleared during communication, ongoing communication is immediately aborted. A restart of the communication is then required.
6	CSIGnTXE	Enables/disables transmission. 0: Transmission disabled 1: Transmission enabled
5	CSIGnRXE	Enables/disables reception. 0: Reception disabled 1: Reception enabled

Cautions

1. When CSIGnPWR = 0, do not change bits 6 (CSIGnTXE), 5 (CSIGnRXE), and 0.
   However, these bits can be changed at the same time that the CSIGnPWR bit changes from 0 to 1.
2. Do not modify CSIGnTXE or CSIGnRXE while a data transmission is pending or ongoing, i.e. if CSIGnSTR0.CSIGnTSF = 1.

(2) CSIGnCTL1 - CSIGn control register 1

This register specifies the interrupt timing and the interrupt delay mode. It enables/disables extended data length control, data consistency check, loop-back mode, handshake function, and slave select function.

Access This register can be read/written in 32-bit units.

Address <CSIGn_base> + 0010_H

Initial Value 0000 0000H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	CSIGnCKR	CSIGnSLIT
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	CSIGnEDLE	0	CSIGnDCS	0	CSIGnLBM	CSIGnSIT	CSIGnHSE	CSIGnSSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIGnCTL0.CSIGnPWR = 0.

Table 24-12 CSIGnCTL1 register contents (1/2)

Bit position	Bit name	Function
17	CSIGnCKR	Selects the CSIGnTSCK clock phase. 0: The CSIGnTSCK default level is high. 1: The CSIGnTSCK default level is low. This bit is used in combination with the CSIGnCFG0.CSIGnDAP bit. For details, see *(7) "CSIGnCFG0 - CSIGn configuration register 0" on page 1822*.
16	CSIGnSLIT	Selects the timing of interrupt CSIGnTIC. 0: Normal interrupt timing (interrupt is generated after the transfer) 1: Interrupt generation when CSIGnTX0W or CSIGnTX0H is free for next data. For details, see *(1) "CSIGnTIC (communication interrupt)" on page 1803*.
7	CSIGnEDLE	Enables/disables extended data length (EDL) mode. 0: Extended data length mode disabled 1: Extended data length mode enabled For details, see *(2) "Data length greater than 16 bits" on page 1799*.
5	CSIGnDCS	Enables/disables data consistency check. 0: Data consistency check disabled 1: Data consistency check enabled For details, see *(1) "Data consistency check" on page 1808*.
3	CSIGnLBM	Controls loop-back mode (LBM). 0: Loop-back mode deactivated 1: Loop-back mode activated For details, see *24.3.10 "Error detection" on page 1808*. This bit cannot be changed in the slave mode.

Table 24-12 CSIGnCTL1 register contents (2/2)

Bit position	Bit name	Function
2	CSIGnSIT	Selects interrupt delay mode. 0: No delay 1: Half clock delay for all interrupts This bit is only valid in master mode. In slave mode, no delay is generated. For details, see *24.3.8 "CSIG interrupts" on page 1803*.
1	CSIGnHSE	Enables/disables handshake mode. 0: Handshake function disabled 1: Handshake function enabled For details, see *24.3.9 "Handshake function" on page 1806*.
0	CSIGnSSE	Enables/disables slave select function (SS). 0: Input signal $\overline{\text{CSIGnTSSI}}$ is ignored 1: Input signal $\overline{\text{CSIGnTSSI}}$ is enabled If the slave select function is not used, this bit must be set to 0 (see also *24.3.2 "Master/slave connections" on page 1795*).

Details about CSIGnCTL1.CSIGnSSE:

Table 24-13 Operation of the slave select function during reception

CSIGnCTL0. CSIGnRXE	CSIGnCTL1. CSIGnSSE	$\overline{\text{CSIGnTSSI}}$	Reception operation
0	-	-	Reception disabled
1	0	-	Possible
1	1	0	Possible
1	1	1	Impossible

Table 24-14 Operation of the slave select function during transmission

CSIGnCTL0. CSIGnTXE	CSIGnCTL1. CSIGnSSE	$\overline{\text{CSIGnTSSI}}$	Transmission operation
0	–	–	Transmission disabled
1	0	–	Possible
1	1	0	Possible
1	1	1	Impossible

(3) CSIGnCTL2 - CSIGn control register 2

This register selects the operating mode, prescaler, and baud rate.

Access This register can be read/written in 16-bit units.

Address <CSIGn_base> + 0014_H

Initial Value $E000_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIGnPRS[2:0]			0	CSIGnBRS[11:0]											
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIGnCTL0.CSIGnPWR = 0.

Table 24-15 CSIGnCTL2 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15 to 13</td><td>CSIGnPRS [2:0]</td><td>Selects the operating mode and the value of the prescaler.
<table>
<tr><th>CSIGn PRS2</th><th>CSIGn PRS1</th><th>CSIGn PRS0</th><th>Base clock (PRSOUT) selection</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>PCLK (master mode)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>PCLK / 2 (master mode)</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>PCLK / 4 (master mode)</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>PCLK / 8 (master mode)</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>PCLK / 16 (master mode)</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>PCLK / 32 (master mode)</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>PCLK / 64 (master mode)</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>External clock via CSIGnTSCK (in) (slave mode)</td></tr>
</table></td></tr>
<tr><td>11 to 0</td><td>CSIGnBRS [11:0]</td><td>Selects the baud rate.
The setting of these bits is valid only in the master mode, and is ignored in the slave mode.
<table>
<tr><th>CSIGnBRS[11:0]</th><th>Baud rate at CSIGnTSCK</th></tr>
<tr><td>0</td><td>BRG is stopped</td></tr>
<tr><td>1</td><td>PCLK / ($2^m \times 1 \times 2$)</td></tr>
<tr><td>2</td><td>PCLK / ($2^m \times 2 \times 2$)</td></tr>
<tr><td>3</td><td>PCLK / ($2^m \times 3 \times 2$)</td></tr>
<tr><td>4</td><td>PCLK / ($2^m \times 4 \times 2$)</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>4095</td><td>PCLK / ($2^m \times 4095 \times 2$)</td></tr>
</table>
Note:
m = 0 to 6: Value specified for CSIGnPRS[2:0]</td></tr>
</table>

(4) CSIGnSTR0 - CSIGn status register 0

This register indicates the status of the CSIG.

Access This register can be read in 32-bit units.

Address <CSIGn_base> + 0004_H

Initial Value $0000\ 0010_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	CSIGnTSF	0	0	1	CSIGnDCE	0	CSIGnPE	CSIGnOVE
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 24-16 CSIGnSTR0 register contents (1/2)

Bit position	Bit name	Function
7	CSIGnTSF	Transfer status flag 0: Idle state 1: Transmission is in progress or being prepared Setting and clearing of this bit is as follows:

Master mode	Set condition	Clear condition
Transmission mode	Writing to transmission data register	Within 0.5 clock cycles from the last CSIGnTSCK edge
Transmission/ reception mode		
Reception mode	Reading from reception data register	

Slave mode	Set condition	Clear condition
Transmission mode	Writing to transmission data register	Within 0.5 clock cycles from the last CSIGnTSCK edge
Transmission/ reception mode		
Reception mode	CSIGnTSCK input	

Table 24-16 CSIGnSTR0 register contents (2/2)

Bit position	Bit name	Function
3	CSIGnDCE	Data consistency error flag 0: No data consistency error detected 1: Data consistency error detected This bit is cleared by setting CSIGnSTCR0.CSIGnDCEC. This bit can be written to when CSIGnCTL0.CSIGnPWR = 0. This bit is initialized when CSIGnCTL0.CSIGnPWR changes from 0 to 1 or from 1 to 0. If this bit is set due to a data consistency error being detected and cleared by CSIGnSTCR0.CSIGnDCEC at the same time, setting the bit is prioritized.
1	CSIGnPE	Parity error flag 0: No parity error detected 1: Parity error detected This bit is cleared by setting CSIGnSTCR0.CSIGnPEC. This bit can be written to when CSIGnCTL0.CSIGnPWR = 0. This bit is initialized when CSIGnCTL0.CSIGnPWR changes from 0 to 1 or from 1 to 0. If this bit is set due to a parity error being detected and cleared by CSIGnSTCR0.CSIGnPEC at the same time, setting the bit is prioritized.
0	CSIGnOVE	Overrun error flag 0: No overrun error detected 1: Overrun error detected This bit is cleared by setting CSIGnSTCR0.CSIGnOVEC. This bit can be written to when CSIGnCTL0.CSIGnPWR = 0. This bit is initialized when CSIGnCTL0.CSIGnPWR changes from 0 to 1 or from 1 to 0. This bit is fixed to 0 in the dual buffer mode. If this bit is set due to an overrun error being detected and cleared by CSIGnSTCR0.CSIGnOVEC at the same time, setting the bit is prioritized.

(5) CSIGnSTCR0 - CSIGn status clear register 0

This register clears the status flags of the CSIGnSTR0 status register.

Access This register can be written in 16-bit units.

When read, the value 0000_H is always returned.

Address <CSIGn_base> + 0008_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	CSIGnDCEC	0	CSIGnPEC	CSIGnOVEC
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 24-17 CSIGnSTCR0 register contents

Bit position	Bit name	Function
3	CSIGnDCEC	Controls the data consistency error flag clear command. 0: No operation. Read value is always 0. 1: Clear data consistency error flag (CSIGnSTR0.CSIGnDCE)
1	CSIGnPEC	Controls the parity error flag clear command. 0: No operation. Read value is always 0. 1: Clear parity error flag (CSIGnSTR0.CSIGnPE)
0	CSIGnOVEC	Controls the overrun error flag clear command. 0: No operation. Read value is always 0. 1: Clear overrun error flag (CSIGnSTR0.CSIGnOVE)

(6) CSIGnBCTL0 - CSIGn reception mode control register 0

This register enables/disables the data transfer in reception mode.

Access This register can be read/written in 8-bit and 1-bit units.

Address <CSIGn_base> + 1000_H

Initial Value 01_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	CSIGnSCE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-18 CSIGnBCTL0 register contents

Bit position	Bit name	Function
0	CSIGnSCE	Disables/enables next data reception start by reading CSIGnRX0. 0: Next reception disabled 1: Next reception enabled For details, see *(2) "Reception Mode" on page 1798* and *24.3.7 "Communication in slave mode" on page 1802*.

Cautions

1. Writing the CSIGnSCE bit must be executed one clock before a CSIGnTIR interrupt is generated.

<R> 2. The CSIGnSCE bits must be fixed to 0 when the operating mode is transmission or transmission/reception mode.

(7) CSIGnCFG0 - CSIGn configuration register 0

This register configures the communication protocol – data length, parity, transfer direction, clock phase, and data phase.

Access This register can be read/written in 32-bit units.

Address <CSIGn_base> + 1010_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	CSIGn PS[1:0]		CSIGn DLS[3:0]				0	0	0	0	0	CSIGn DIR	0	CSIGn DAP
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

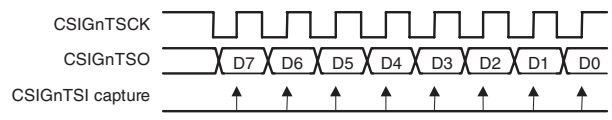
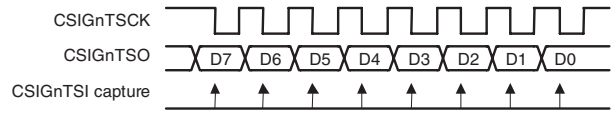
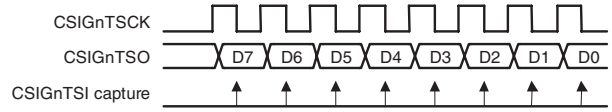
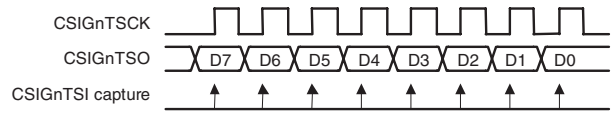
Caution Writing is only possible when CSIGnCTL0.CSIGnPWR = 0 (except when writing the same value, which is possible even when CSIGnCTL0.CSIGnPWR = 1).

Table 24-19 CSIGnCFG0 register contents (1/2)

Bit position	Bit name	Function
29 and 28	CSIGnPS [1:0]	Specifies the parity.

CSIGn PS1	CSIGn PS0	Transmission	Reception
0	0	No parity transmitted	Parity reception is not expected.
0	1	Add parity bit fixed at 0	Parity bit reception is expected, but parity judgment is not performed.
1	0	Add odd parity	Odd parity bit reception is expected.
1	1	Add even parity	Even parity bit reception is expected.

Table 24-19 CSIGnCFG0 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>27 to 24</td><td>CSIGnDLS [3:0]</td><td>Selects the data length.
<table>
<tr><th>CSIGnDLS[3:0]</th><th>Data length</th></tr>
<tr><td>0000_B</td><td>16 bits</td></tr>
<tr><td>0001_B</td><td>1 bit</td></tr>
<tr><td>0010_B</td><td>2 bits</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1111_B</td><td>15 bits</td></tr>
</table>
Note:
Data length between 1 bit and 6 bits requires that EDL function is used (see also (2) “Data length greater than 16 bits” on page 1799). In addition, it is forbidden to transmit two consecutive data with a length of less than 7 bits.
The setting of these bits is valid when CSIGnTX0W.CSIGnEDL[3:0] = 0.</td></tr>
<tr><td>18</td><td>CSIGnDIR</td><td>Selects the serial data direction.
0: Data is sent/received with MSB first
1: Data is sent/received with LSB first</td></tr>
<tr><td>16</td><td>CSIGnDAP</td><td>CKR: Clock phase selection bit
DAP: Data phase selection bit
<table>
<tr><th>CSIGn CKR</th><th>CSIGn DAP</th><th>Clock and data phase selection</th></tr>
<tr><td>0</td><td>0</td><td> 
</td></tr>
<tr><td>0</td><td>1</td><td> 
</td></tr>
<tr><td>1</td><td>0</td><td> 
</td></tr>
<tr><td>1</td><td>1</td><td> 
</td></tr>
</table>
For details about the CSIGnCKR bit, see (2) “CSIGnCTL1 - CSIGn control register 1” on page 1815.</td></tr>
</table>

(8) CSIGnTX0W - Transmission data register 0 for word access

This register stores the transmission data. It has to be used when the extended data length function is enabled (CSIGnCTL1.SCIGnEDLE = 1).

Access This register can be read/written in 32-bit units.

Address <CSIGn_base> + 1004_H

Initial Value Undefined

Caution Writing to this register is prohibited when CSIGnCTL0.CSIGnTXE and CSIGnCTL0.CSIGnRXE = 0.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	CSIGnEDL	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R/W	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIGnTX[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-20 CSIGnTX0W register contents

Bit position	Bit name	Function
29	CSIGnEDL	Specifies whether the associated data requires the extended data length (EDL) option. 0: Normal operation 1: Extended data length activated The associated data is transmitted as of 16 bits. The inter-data delay time and idle time are not inserted after data transmission. When CSIGnCTL1.CSIGnEDLE = 1 and CSIGnTX0W.CSIGnEDL = 1, the same slave must also be selected for the second data. If the slave for the second data is changed, the correct operation is not guaranteed. **Caution:** This bit can only be used when CSIGnCTL1.CSIGnEDLE = 1.
15 to 0	CSIGn TX[15:0]	Stores the transmission data.

(9) CSIGnTX0H - Transmission data register 0 for half word access

This register stores the transmission data. It can be used when the Extended Data Length function is disabled (CSIGnCTL1.CSIGnEDLE = 0).

Access This register can be read/written in 16-bit units.

Address <CSIGn_base> + 1008_H

Initial Value 0000_H. This register is initialized by any reset.

Caution Writing to this register is prohibited when CSIGnCTL0.CSIGnTXE and CSIGnCTL0.CSIGnRXE = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIGnTX[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-21 CSIGnTX0H register contents

Bit position	Bit name	Function
15 to 0	CSIGn TX[15:0]	Stores the transmission data.

(10) CSIGnRX0 - Reception data register 0

This register stores the received data.

Access This register can be read in 16-bit units.

<R> **Address** <CSIGn_base> + $100C_H$

Initial Value 0000_H. This register is initialized by any reset.

Cautions

1. This register can be read when CSIGnCTL0.CSIGnPWR = 1, and can be written to when CSIGnCTL0.CSIGnPWR = 0.
2. This register is initialized when the value of CSIGnCTL0.CSIGnPWR changes.
3. Reading this register is prohibited when CSIGnCTL0.CSIGnTXE and CSIGnCTL0.CSIGnRXE = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIGnRX[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 24-22 CSIGnRX0 register contents

Bit position	Bit name	Function
15 to 0	CSIGn RX[15:0]	Stores the reception data.

24.5 Operating Procedure Example

(1) For transmission/reception in the master mode

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIGnCFG0.CSIGnDLS[3:0] = 1000_B)
- Transmission direction: MSB first (CSIGnCFG0.CSIGnDIR = 0)
- Normal clock phase and data phase (CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0)
- No delay for any interrupt (CSIGnCTL1.CSIGnSIT = 0)
- A CSIGnTIC interrupt is generated when transferring starts. (CSIGnCTL1.CSIGnCLIT = 1)

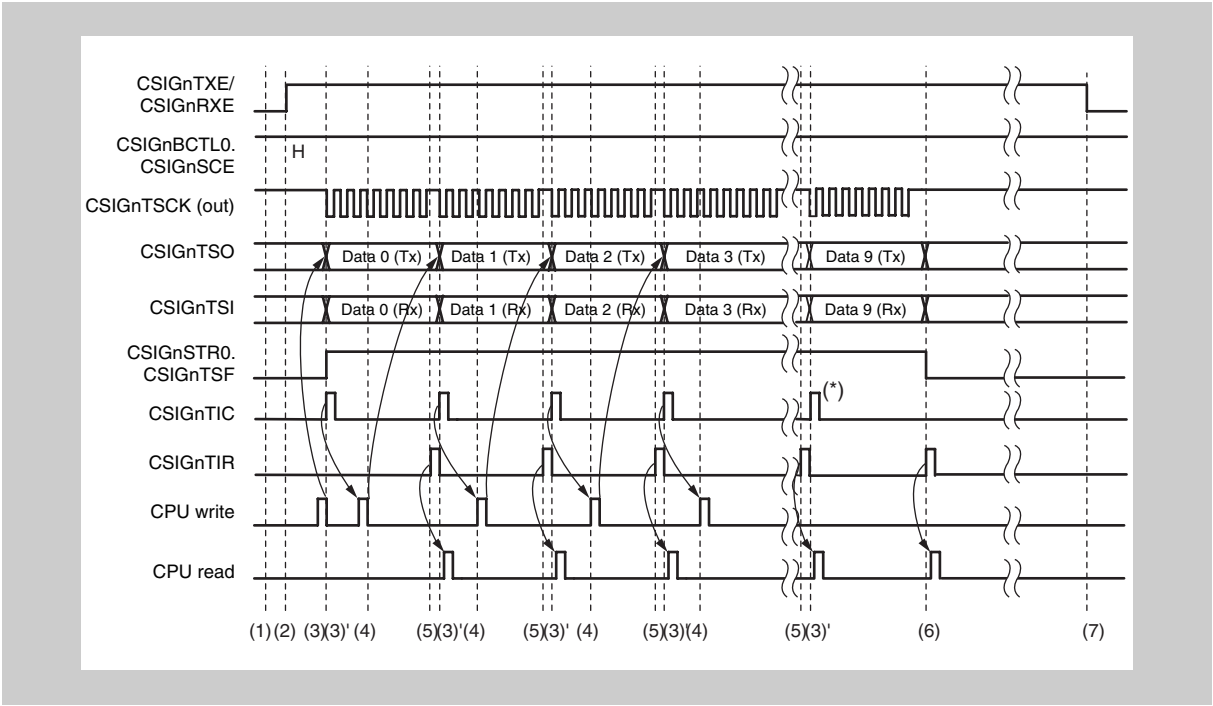


Figure 24-26 For transmission/reception in the master mode

Procedure:

1. Set up the following registers before setting CSIGnCTL0.CSIGnPWR to 1: CSIGnCTL1, CSIGnCTL2, CISGnBCTL0, CSIGnCFG0
2. CSIGnCTL0.CSIGnPWR = 1 (clock enabled)
   CSIGnCTL0.CSIGnTXE = 1 (transmission enabled)
   CSIGnCTL0.CSIGnRXE = 1 (reception enabled)
   Bit 0 of CSIGnCTL0 = 1
3. Write the first data to the transmission data register CSIGnTX0W. Transmission automatically starts.

3'. When CSIGnCTL1.CSIGnSLIT is set to 1, CSIGnTIC is generated at the start edge of CSIGnTSCK. CSIGnTIC indicates that the second data can be written to CSIGnTX0W.

4. Write the second data to CSIGnTX0W. By writing the second data immediately after writing the first data, the unnecessary inter-data delay can be avoided.

5. Each time data is received, a CSIGnTIR interrupt is generated.
   – CSIGnTIR indicates that the reception data register CSIGnRX0 must be read.
6. If the CSIGnTIC interrupt indicated by “*” in the figure is the last one, it is not necessary to write to the transmission data register CSIGnTX0W based on the corresponding CSIGnTIC interrupt.
7. Finally, clear CSIGnCTL0.CSIGnTXE and CSIGnCTL0.CSIGnRXE to disable transmission/reception operations. In addition, clear CSIGnCTL0.CSIGnPWR to reduce the power consumption of the CSIG.

(2) For reception in the master mode

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIGnCFG0.CSIGnDLS[3:0] = 1000_B)
- Transmission direction: MSB first (CSIGnCFG0.CSIGnDIR = 0)
- Normal clock phase and data phase (CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0)
- No delay for any interrupt (CSIGnCTL1.CSIGnSIT = 0)
- A CSIGnTIC interrupt is generated when transferring starts. (CSIGnCTL1.CSIGnCLIT = 1)

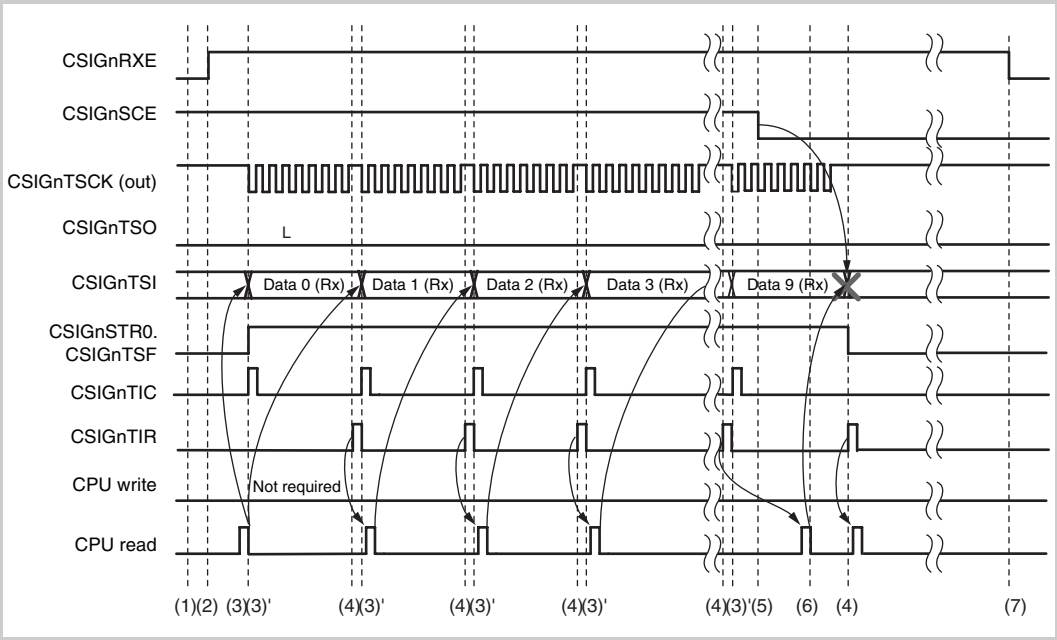


Figure 24-27 For reception in the master mode

Procedure:

1. Set up the following registers before setting CSIGnCTL0.CSIGnPWR to 1: CSIGnCTL1, CSIGnCTL2, CISGnBCTL0, CSIGnCFG0
2. CSIGnCTL0.CSIGnPWR = 1 (clock enabled)
   CSIGnCTL0.CSIGnTXE = 0 (transmission disabled)
   CSIGnCTL0.CSIGnRXE = 1 (reception enabled)
   Bit 0 of CSIGnCTL0 = 1
3. Read the dummy data from the reception data register CSIGnRX0. Reception automatically starts.

3'. When CSIGnCTL1.CSIGnSLIT is set to 1, CSIGnTIC is generated at the start edge of CSIGnTSCK.

4. Each time data is received, a CSIGnTIR interrupt is generated.
   – CSIGnTIR indicates that the reception data register CSIGnRX0 must be read.
5. To finish the consecutive reception with the data currently received, clear the CSIGnBCTL0.CSIGnSCE bit.

6. Reception does not start even if reception data is read.
7. Finally, clear CSIGnCTL0.CSIGnRXE to disable reception operations. In addition, clear CSIGnCTL0.CSIGnPWR to reduce the power consumption of the CSIG.

(3) For transmission/reception in the slave mode

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIGnCFG0.CSIGnDLS[3:0] = 1000_B)
- Transmission direction: MSB first (CSIGnCFG0.CSIGnDIR = 0)
- Normal clock phase and data phase (CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0)
- No delay for any interrupt (CSIGnCTL1.CSIGnSIT = 0)
- A CSIGnTIC interrupt is generated when transferring starts. (CSIGnCTL1.CSIGnCLIT = 1)

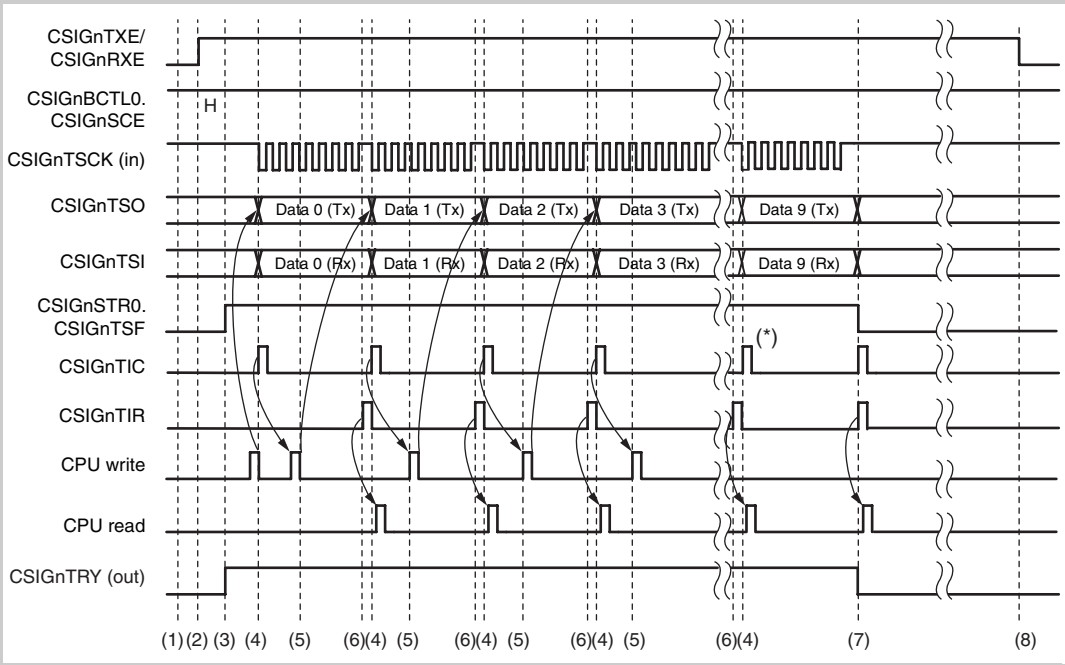


Figure 24-28 For transmission/reception in the slave mode

Procedure:

1. Set up the following registers before setting CSIGnCTL0.CSIGnPWR to 1: CSIGnCTL1, CSIGnCTL2, CISGnBCTL0, CSIGnCFG0
2. CSIGnCTL0.CSIGnPWR = 1 (clock enabled)
   CSIGnCTL0.CSIGnTXE = 1 (transmission enabled)
   CSIGnCTL0.CSIGnRXE = 1 (reception enabled)
   Bit 0 of CSIGnCTL0 = 1
3. Write the first data to the transmission data register CSIGnTX0W.
4. When a serial clock is supplied from the master, transmission automatically starts.
   When CSIGnCTL1.CSIGnSLIT is set to 1, CSIGnTIC is generated at the start edge of CSIGnTSCK. CSIGnTIC indicates that the second data can be written to CSIGnTX0W.
5. Write the second data to CSIGnTX0W. By writing the second data immediately after writing the first data, the unnecessary inter-data delay can be avoided.

6. Each time data is received, a CSIGnTIR interrupt is generated.
   – CSIGnTIR indicates that the reception data register CSIGnRX0 must be read.
7. If the CSIGnTIC interrupt indicated by “*” in the figure is the last one, it is not necessary to write to the transmission data register CSIGnTX0W based on the corresponding CSIGnTIC interrupt.
8. Finally, clear CSIGnCTL0.CSIGnTXE and CSIGnCTL0.CSIGnRXE to disable transmission/reception operations. In addition, clear CSIGnCTL0.CSIGnPWR to reduce the power consumption of the CSIG.

(4) For reception in the slave mode

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIGnCFG0.CSIGnDLS[3:0] = 1000_B)
- Transmission direction: MSB first (CSIGnCFG0.CSIGnDIR = 0)
- Normal clock phase and data phase (CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0)
- No delay for any interrupt (CSIGnCTL1.CSIGnSIT = 0)
- A CSIGnTIC interrupt is generated when transferring starts. (CSIGnCTL1.CSIGnCLIT = 1)

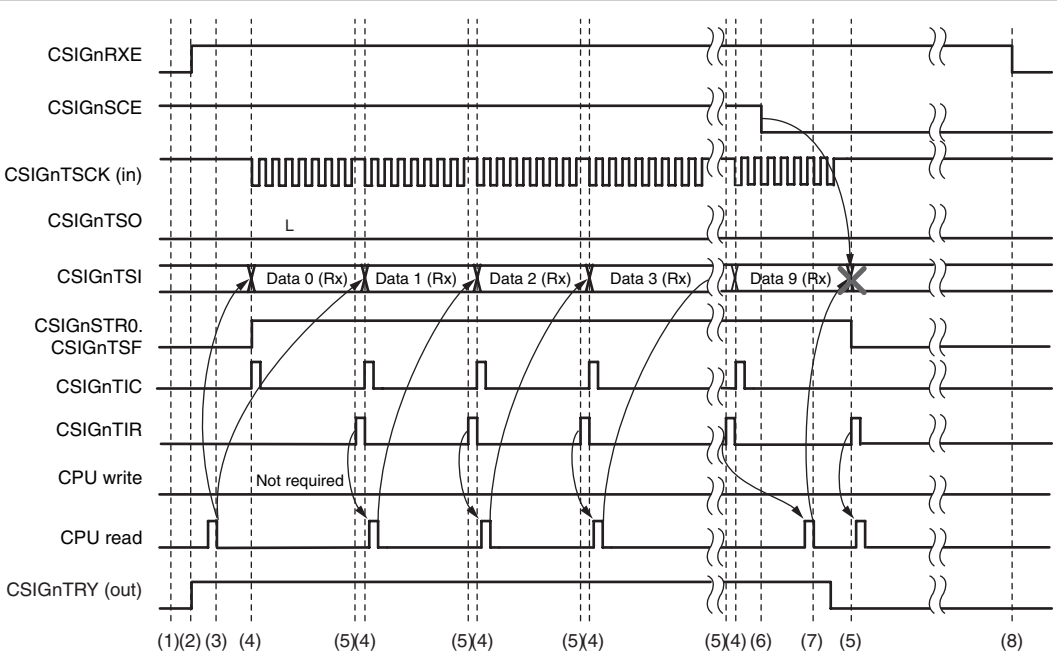


Figure 24-29 For reception in the slave mode

Procedure:

1. Set up the following registers before setting CSIGnCTL0.CSIGnPWR to 1: CSIGnCTL1, CSIGnCTL2, CISGnBCTL0, CSIGnCFG0
2. CSIGnCTL0.CSIGnPWR = 1 (clock enabled)
   CSIGnCTL0.CSIGnTXE = 0 (transmission disabled)
   CSIGnCTL0.CSIGnRXE = 1 (reception enabled)
   Bit 0 of CSIGnCTL0 = 1
3. Read the dummy data from the reception data register CSIGnRX0. Reception automatically starts.
4. When a serial clock is supplied from the master, transmission automatically starts.
   When CSIGnCTL1.CSIGnSLIT is set to 1, CSIGnTIC is generated at the start edge of CSIGnTSCK.
5. Each time data is received, a CSIGnTIR interrupt is generated.
   – CSIGnTIR indicates that the reception data register CSIGnRX0 must be read.

6. To finish the consecutive reception with the data currently received, clear the CSIGnBCTL0.CSIGnSCE bit.
7. Reception does not start even if reception data is read.
8. Finally, clear CSIGnCTL0.CSIGnRXE to disable reception operations. In addition, clear CSIGnCTL0.CSIGnPWR to reduce the power consumption of the CSIG.

Chapter 25 Clocked Serial Interface H (CSIH)

This chapter describes clocked serial interface H (CSIH).

The first section describes all V850E2/Sx4-H specific properties, such as instances, register base addresses, and input/output signal names.
The subsequent sections describe the features that apply to all implementations.

25.1 V850E2/Sx4-H CSIH Features

Instances

This microcontroller has following number of instances of CSIH:

Table 25-1 Instances of CSIH

CSIH	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of instances	2	3	3
Name	CSIH0, CSIH1	CSIH0 to CSIH2	CSIH0 to CSIH2

Instances index n

Throughout this chapter, the individual instances of CSIH is identified by the index "n" (n = 0 to 2), for example, CSIHnCTL0 for CSIHn control register 0.

Chip select index x

CSIH has up to 8 chip select signals. Throughout this chapter, the individual chip select signals are identified by the index “x” (x = 0 to 7), thus a certain chip select signal is denoted as CSx.
The number of chip selects for each instance of CSIH is given in the following table:

Table 25-2 CSIH chip select indices

CSIHn	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
CSIH0	x = 0 to 3	x = 0 to 3	x = 0 to 7
CSIH1	x = 0 to 3	x = 0 to 3	x = 0 to 7
CSIH2	–	x = 0 to 7	x = 0 to 7

Register addresses

All CSIHn register addresses are given as addresses offset from the individual base address <CSIHn_base>.
The base address <CSIHn_base> of each CSIHn is listed in the following table:

Table 25-3 Register base addresses <CSIHn_base>

CSIHn	<CSIHn_base> address
CSIH0	FF6C 0000_H
CSIH1	FF6D 0000_H
CSIH2	FF6E 0000_H

Clock supply

The following clock is supplied to CSIH:

Table 25-4 CSIHn clock supply

CSIHn	Clock	Connected to:
CSIH0	PCLK	Clock generator CKSCLK_109
CSIH1	PCLK	Clock generator CKSCLK_109
CSIH2	PCLK	Clock generator CKSCLK_109

Maximum transfer speed (baud rate)

For CSIH, communication is possible at the maximum transfer speeds (baud rates) listed in the following table.

Table 25-5 Maximum CSIHn transfer speeds (baud rates)

Mode	Maximum transfer speed (baud rate)
Master mode	10 MHz
Slave mode	8 MHz

Interrupts and DMA

CSIH can generate the following interrupt requests and DMA requests:

Table 25-6 CSIHn interrupt requests and DMA requests

CSIHn signals	Function	Connected to:
CSIH0:		
CSIH0TIC	Communication status interrupt	Interrupt controller INTCSIH0IC DMA controller trigger 77
CSIH0TIR	Reception status interrupt	Interrupt controller INTCSIH0IR DMA controller trigger 81
CSIH0TIRE	Communication error interrupt	Interrupt controller INTCSIH0IRE
CSIH0TIJC	Job completion interrupt	Interrupt controller INTCSIH0IJC DMA controller trigger 78
CSIH1:		
CSIH1TIC	Communication status interrupt	Interrupt controller INTCSIH1IC DMA controller trigger 88
CSIH1TIR	Reception status interrupt	Interrupt controller INTCSIH1IR DMA controller trigger 87
CSIH1TIRE	Communication error interrupt	Interrupt controller INTCSIH1IRE
CSIH1TIJC	Job completion interrupt	Interrupt controller INTCSIH1IJC DMA controller trigger 89
CSIH2:		
CSIH2TIC	Communication status interrupt	Interrupt controller INTCSIH2IC DMA controller trigger 97
CSIH2TIR	Reception status interrupt	Interrupt controller INTCSIH2IR DMA controller trigger 96
CSIH2TIRE	Communication error interrupt	Interrupt controller INTCSIH2IRE
CSIH2TIJC	Job completion interrupt	Interrupt controller INTCSIH2IJC DMA controller trigger 98

CSIH hardware reset

CSIH and its registers are initialized by the following reset signal:

Table 25-7 CSIHn reset signal

CSIHn	Reset signal
CSIHn	System reset SYSRES

I/O signals

The I/O signals of CSIH are listed in the following table:

Table 25-8 CSIHn I/O signals

CSIHn signal	Function	Connected to:
CSIH0:		
CSIH0TSCK	Serial clock signal	Port CSIH0SC
CSIH0TSI	Serial data input signal	Port CSIH0SI
CSIH0TSO	Serial data output signal	Port CSIH0SO
CSIH0TSSI	Slave select input signal	Port $\overline{\text{CSIH0SSI}}$
CSIH0TRY	Ready/busy input signal CSIH0RYI	Port CSIH0RYI
	Ready/busy input signal CSIH0RYO	Port CSIH0RYO
CSIHnTCSSn[7:0]	Chip select signals	Port CSIH0CSS[7:0]
CSIH1:		
CSIH1TSCK	Serial clock signal	Port CSIH1SC
CSIH1TSI	Serial data input signal	Port CSIH10SI
CSIH1TSO	Serial data output signal	Port CSIH1SO
CSIH1TSSI	Slave select input signal	Port $\overline{\text{CSIH1SSI}}$
CSIH1TRY	Ready/busy signal	Port CSIH1RYI
CSIHnTCSSn[7:0]	Chip select signals	Port CSIH1CSS[7:0]
CSIH2:		
CSIH2TSCK	Serial clock signal	Port CSIH2SC
CSIH2TSI	Serial data input signal	Port CSIH2SI
CSIH2TSO	Serial data output signal	Port CSIH2SO
CSIH2TSSI	Slave select input signal	Port $\overline{\text{CSIH2SSI}}$
CSIH2TRY	Ready/busy signal	Port CSIH2RYI
CSIH2TCSSn[7:0]	Chip select signals	Port CSIH2CSS[7:0]

<R> **Caution** Port filters are assigned to the input pins of clocked serial interface H (CSIHn), and these filters are enabled as the initial setting. However, because the use of the port filters might cause a communication error, do not use the port filters when using CSIHn; instead, enable the filter bypass by setting the corresponding registers as shown below.

CSIH0SC: FLCA22CTL0 = 80H, CSIH0RY: FLCA22CTL1 = 80H,
CSIH0SI: FLCA22CTL2 = 80H, $\overline{\text{CSIH0SSI}}$: FLCA22CTL3 = 80H,
CSIH1SC: FLCA22CTL4 = 80H, CSIH1RY: FLCA22CTL5 = 80H,
CSIH1SI: FLCA22CTL6 = 80H, $\overline{\text{CSIH1SSI}}$: FLCA22CTL7 = 80H,
CSIH2SC: FLCA23CTL0 = 80H, CSIH2RY: FLCA23CTL1 = 80H,
CSIH2SI: FLCA23CTL2 = 80H, $\overline{\text{CSIH2SSI}}$: FLCA23CTL3 = 80H

Data consistency check

The following table shows the CSIHnSO ports and their capability to use them for data consistency checks. For details about data consistency checks, see *25.3.14 “Error detection”*.

Table 25-9 CSIHn data consistency check ports

CSIHn I/O port signal	Port	Alternative function	Data consistency check
CSIH0:			
CSIH0SO	P1_7	ALT_OUT4	Possible
	P25_2	ALT_OUT4	Possible
CSIH1:			
CSIH1SO	P0_5	ALT_OUT3	Not possible
	P25_10	ALT_OUT4	Possible
CSIH2:			
CSIH2SO	P26_2	ALT_OUT4	Possible

25.2 Functional Overview

Features summary

- Three-wire serial synchronous data transfer
- Master mode and slave mode selectable
- Multiple slaves configuration plus RCB (Recessive Configuration for Broadcasting) thanks to eight configurable chip select output signals
- Slave select input signal ($\overline{\text{CSIHnTSSI}}$)
- Built-in baud rate generator
- Baud rate adjustable; in slave mode determined by input clock
- Maximum transmission speed:
  - in master mode: PCLK/4
  - in slave mode: PCLK/6

Caution There might be restrictions on the maximum baud rate that can actually be used depending on the product. Specify the baud rate so as not to exceed the maximum rate for each product.

- Phase of clock and data selectable
- Data transfer with MSB or LSB first selectable
- Transfer data length selectable from 7 to 16 bits in 1-bit units
- Extended data length (EDL) function for transferring data with more than 16 bits
- Three selectable transfer modes:
  - transmit-only mode
  - receive-only mode
  - transmit/receive mode
- Built-in handshake function
- Error detection (data consistency check, parity, timeout, overflow, overrun)
- Full support of job concept
- 128 words I/O buffer memory
- Memory mode selectable (FIFO, dual buffer, Tx-only buffer, direct access)
- Four different interrupt request signals (CSIHnTIC, CSIHnTIR, CSIHnTIRE, CSIHnTIJC)
- Loop back mode (LBM) function for self test

The block diagram shows the main components of the CSIH.

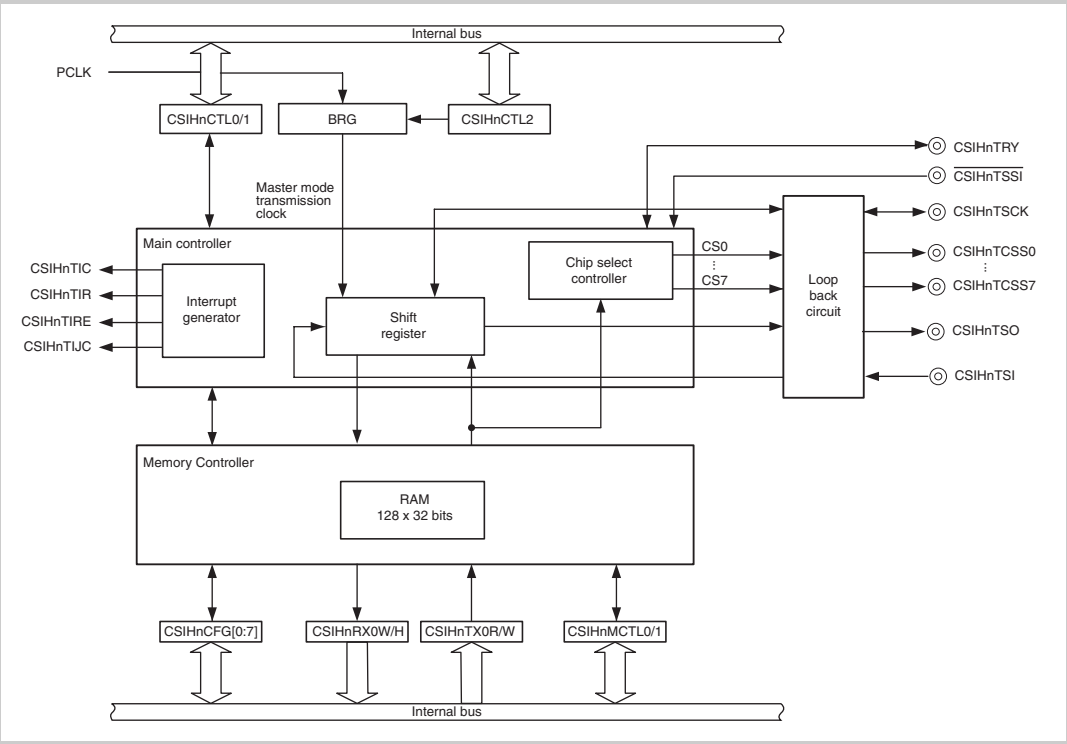


Figure 25-1 CSIH block diagram

In the master mode, the serial clock CSIHnTSCK is generated by the internal baud rate generator (BRG). In the slave mode, the serial clock is supplied from an external source.

The built-in memory can be configured as FIFO, dual buffer (separate transmit and receive buffers), or transmit-only buffer. It can also be bypassed for data transmission and reception without buffering.

The loop back circuit disconnects the CSIH completely from the ports and supports internal self test.

Note This chapter describes the following modes:

- The “operating mode” separates between master and slave mode. In this context, only a master can control and communicate with several slaves (for details, see *25.3.1 “Operating modes (master/slave)” on page 1842*).
- The “job mode” is related to the Autosar job concept (for details, see *25.3.4 “Chip select timing details” on page 1848*).
- The “memory mode” takes the various configurations of the associated buffer memory into account (for details, see *25.3.7 “CSIH buffer memory” on page 1853*).
- The “data transfer mode” specifies the kind of the communication – transmit-only, receive only, or both (for details, see *25.3.8 “Data transfer modes” on page 1855*).

25.3 Functional Description

The Clocked Serial Interface H uses three signals for communication:

- Serial clock CSIHnTSCK (output for the master mode or input for the slave mode)
- Data output signal CSIHnTSO
- Data input signal CSIHnTSI

Additional signals are available for external control and monitoring.

- $\overline{\text{CSIHnTSSI}}$: Slave select input signal
- CSIHnTRY: Handshake signal (input for the master mode or output for the slave mode)
- CSIHnTCSS[7:0]: Chip select signal

Data transmission is bit-wise and serial and synchronous to the serial clock.

The most important registers for setting up the CSIH are:

Register	Function
CSIHnCTL0	Enables or disables the operation clock (PCLK) and enables or disables data transmission and reception. Defines end-of-job behavior and enables/disables buffering (bypass of the buffer).
CSIHnCTL1	Controls options like interrupt timing, extended data length, job feature, data consistency check, loop-back mode, handshake, etc.
CSIHnCTL2	Selects master/slave mode and – effective in master mode – the baud rate of the Internal baud rate generator (BRG)
CSIHnMCTL0	Selects memory mode and specifies timeout
CSIHnMCTL1	Controls the memory in FIFO mode
CSIHnMCTL2	Controls the memory in the dual buffer mode or transmit-only buffer mode
CSIHnCFGx	Registers to configure the communication protocol for each chip select signal

25.3.1 Operating modes (master/slave)

Master/slave selection is performed by using the CSIHnCTL2.CSIHnPRS[2:0] bits, and, when the master is selected, the source clock of the transmission clock must also be selected.

(1) Master mode

In the master mode, the serial clock is generated by the internal baud rate generator (BRG) and provided to the slave(s) by signal CSIHnTSCK.

Master mode is enabled by setting CSIHnCTL2.CSIHnPRS[2:0] to anything but 111_B. In the master mode, the BRG frequency can be specified by specifying values for the CSIHnCTL2.CSIHnPRS[2:0] and CSIHnCTL2.CSIHnBRS[11:0] bits in combination.

Chip select signals

In master mode, one or several chip select signals can be used. If several slaves are connected to the master, the chip select signals can be used to address one or several of the slaves. Only a selected slave is then enabled for communication.

The communication protocol as well as additional parameters are stored separately for each chip select signal. This makes it possible to adapt the data transfer individually to the requirements of each slave. For details, see *25.3.3 “Chip selection (CS) features” on page 1846*.

Clock defaults

The default level of CSIHnTSCK depends on the clock phase selection bit: It is high when CSIHnCFGx.CSIHnCKPx = 0, and is low when CSIHnCFGx.CSIHnCKPx = 1.

The example below shows the communication in master mode for 8 data bits, CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0, and MSB first:

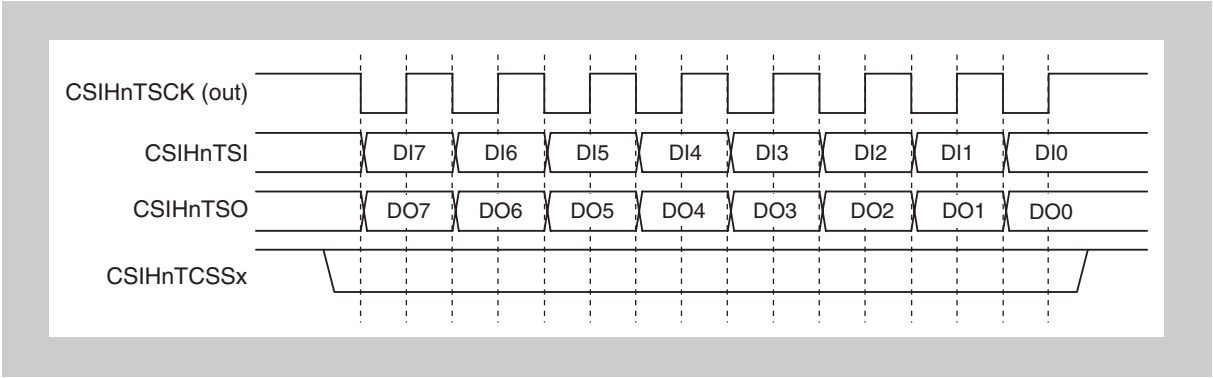


Figure 25-2 Transmit/receive in master mode

(2) Slave mode

In the slave mode, another device is the communication master. The serial clock is supplied through the CSIHnTSCK signal. When the serial clock signal is detected, a transmission or reception operation immediately starts.

Slave mode is selected by setting CSIHnCTL2.CSIHnPRS[2:0] to 111_B.

In slave mode, the transmission protocol setting of the CSIHnCFG0 register are relevant (The CSIHnCFG1 to CSIHnCFG7 register settings become invalid.):

- CSIHnPS0: Parity usage
- CSIHnDLS0: Data length selection
- CSIHnDIR0: Data direction
- CSIHnCKP0, CSIHnDAP0: Clock phase and data phase

Note When using the slave mode, the baud rate generator (BRG) can be disabled by clearing the CSIHnCTL2.CSIHnBRS[11:0] bits, reducing power consumption. However, when using the timeout error function, the BRG must be set to a value other than 0.

The example below shows the communication in the save mode for eight data bits when CSIHnCTL1.CSIHnCKR = 0, CSIHnCFGx.CSIHnDAPx = 0, and the MSB is first.

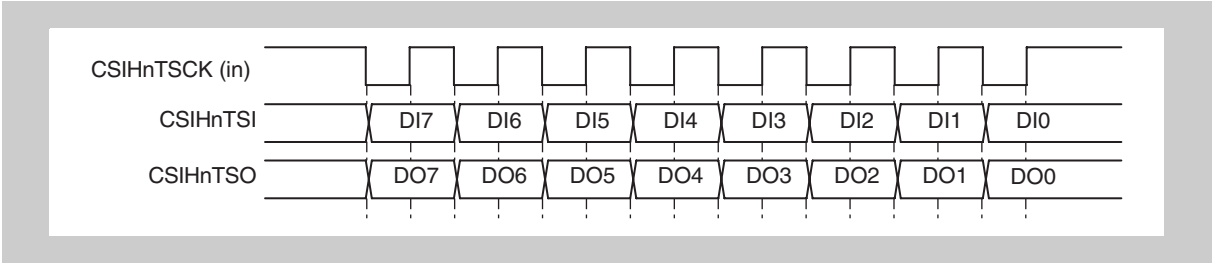


Figure 25-3 Transmission/reception in the slave mode

25.3.2 Master/slave connections

(1) One master and one slave

The following figure illustrates the connections between one master and one slave.

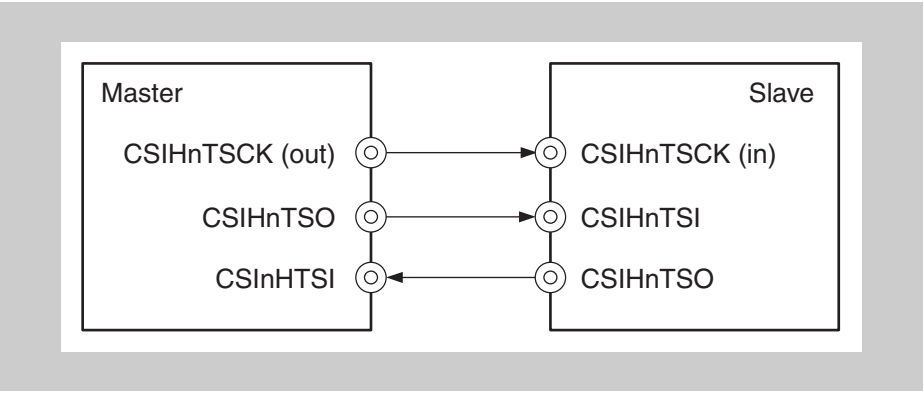


Figure 25-4 Direct master/slave connection

(2) One master and multiple slaves

The following figure illustrates the connections between one master and multiple slaves. In this example, a configuration in which the master supplies one chip select (CS) signal to each slave is possible. This signal is connected to the slave select input $\overline{\text{CSIHnTSSI}}$ of the slave.

The $\overline{\text{CSIHnTSSI}}$ signal recognition function can be enabled/disabled by bit CSIHnCTL1.CSIHnSSE.

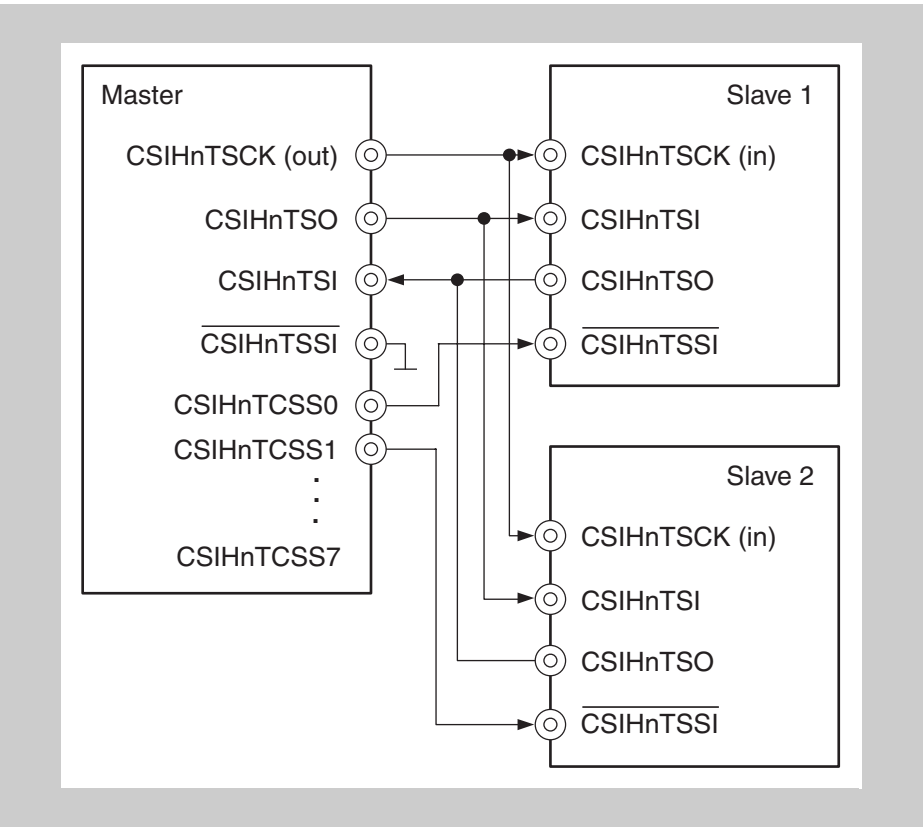


Figure 25-5 Master to multiple slaves connection

The default chip select level is active low. In other words, when the slave select input signal ($\overline{\text{CSIHnTSSI}}$) of a slave is at the low level, that slave is selected as a CSIH slave (and enabled). However, to use a chip select signal (CS) for another device, programming that sets the chip select signal output level to active high is possible.

If a slave is not selected, it will neither receive nor transmit data. In addition, its output CSIHnTSO is set to input mode in order to avoid interference with the output of another slave that was selected.

(3) CSIHnTSO output control

The CSIH can output CSIHnTSO when all of the following conditions are satisfied:

- The CSIH is enabled (CSIHnCTL0.CSIHnPWR = 1).
- The CSIH is operated in transmit-only or transmit/receive mode (CSIHnCTL0.CSIHnTXE = 1).
- The CSIH is operated with slave select enabled (CSIHnCTL1.CSIHnSSE = 1).
- The slave mode selection signal $\overline{\text{CSIHnTSSI}}$ is inactive, i.e. on high level.

By this signal congestions on the external CSIHnTSO signal line are avoided.

25.3.3 Chip selection (CS) features

The chip select signals CSIHnTCSSx can be used by the master to select one or several slaves for communication.

(1) Configuration registers

The parameters for each chip select signal CSIHnTCSSx are defined in the corresponding configuration register CSIHnCFGx. The parameters include the communication protocol and additional CS parameters.

The communication protocol specifies:

- Data length: The number of bits to be sent or received.
- Transfer direction: MSB or LSB first.
- Parity usage: Odd, even, 0 parity, or none.
- Clock phase and data phase.

Additional parameters for each chip select and only available in master mode are:

- Prescaler selection of the baudrate generator separately for each chip select
- Chip select priority: Separates between “dominant” and “recessive” chip select signals. The priority applies if two or more chip selects with different configurations are simultaneously activated for message broadcasting. In this case, the configuration that is set as dominant is used.

  The principle is also called “Recessive Configuration for Broadcasting” (RCB).

Caution It is forbidden to specify several chip select signals as dominant with different configurations unless all dominant chip selects have the same configuration.

- Chip select timing:
  - Setup time T_{setup}: The time from setting the CS signal active to starting data output.
  - Inter-data time T_{inter}: The time between data while the same CS signal is active.
  - Hold time T_{hold}: Hold time of CS active level before changing the CS.
  - Idle time T_{idle}: Inactive time after terminating a CS signal or after every data transfer to the same CSx.

The figure below shows the timing of the chip select (CSx) signal setup time, inter-data time, hold time, and idle time. No matter which CSIHnCFGx.CSIHnIDLx bit is set (to 1), idle time is added to all CS segments.

Figure 25-6 “Chip select timings” shows an example in which the default active low setting is specified for the CS1 and CS2 signals (CSIHnCTL1.CSIHnCSL1 = 0, CSIHnCTL1.CSIHnCSL2 = 0). The active level can be separately specified for each CS.

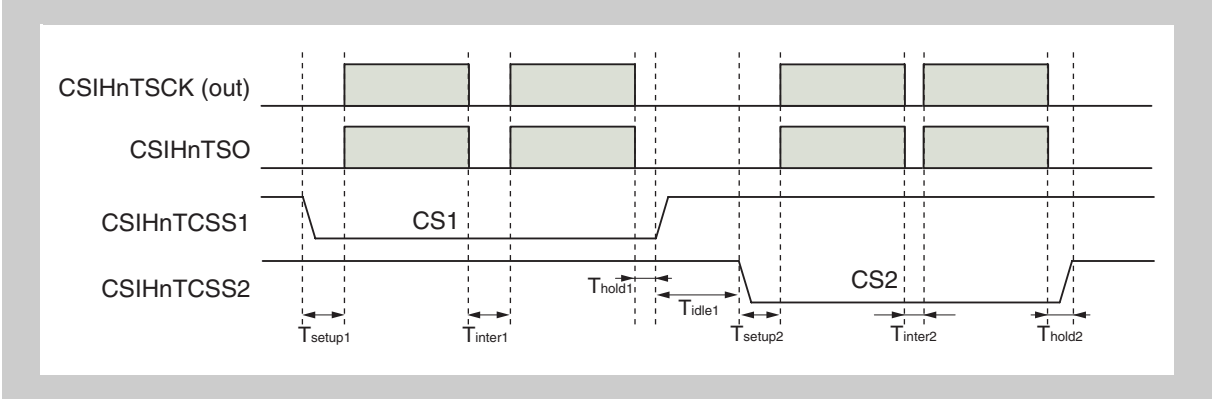


Figure 25-6 Chip select timings

Note that each CS can have a different value for setup, inter-data gap, hold and idle periods

A particular chip select signal is activated by setting the appropriate bit in the transmission data register CSIHnTX0W.CSIHnCS[7:0].

The reception data register indicates in CSIHnRX0W.CSIHnCS[7:0] the chip select signal associated with the received data.

(2) CS example

The following figure shows an example of two consecutive transmissions.

The first communication uses CS0 to address one single slave. The second (for which communication is performed using the dominant-side communication settings) enables CS0 and CS1 to broadcast a message to two slaves. The priority of CS0 is set to “recessive: low priority”, the priority of CS1 to “dominant: high priority”.

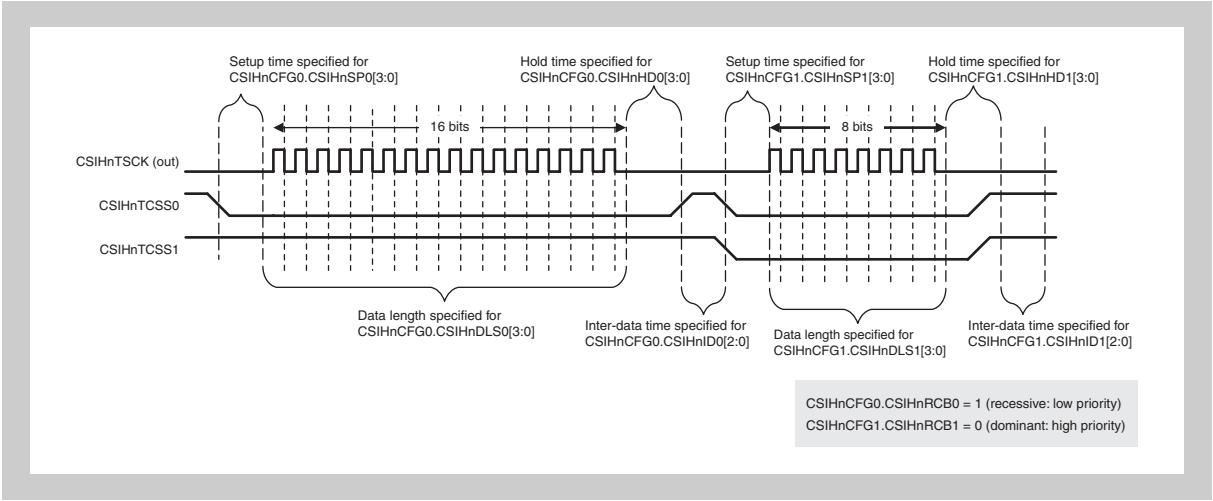


Figure 25-7 Chip select and RCB example

25.3.4 Chip select timing details

(1) Changing the clock phase

The serial clock level is specified for each chip select according to CSIHnCFGx.CSIHnCKPx. The chip select or serial clock level is switched during the idle time. The minimum idle time is 1/2 of a serial clock (CSIHnTSCK) cycle (0.5 SCK).

If the idle time is set to 0.5 transmission clock cycles (in CSIHnCFGx.CSIHnIDx[2:0]) and two consecutive data are sent with different CSIHnCFGx.CSIHnCKPx configuration, the idle time is automatically extended to one cycle of CSIHnTSCK.

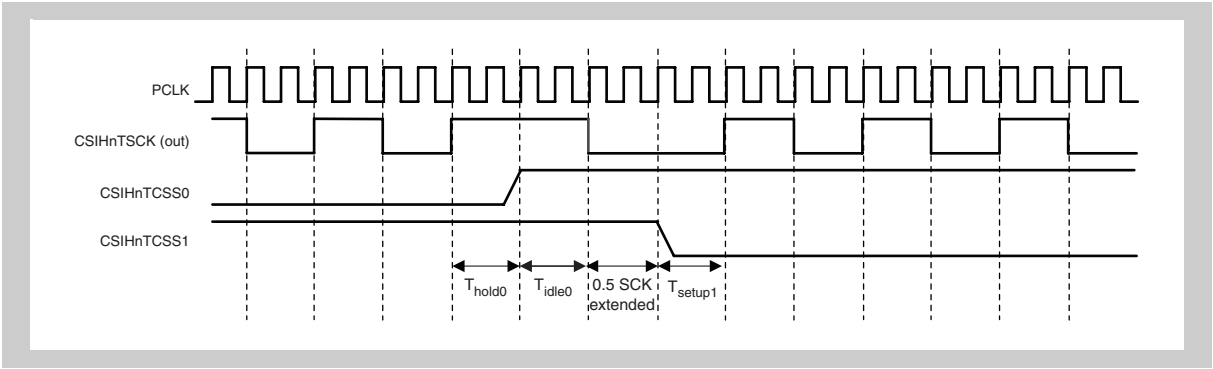


Figure 25-8 Clock phase timing (in the case of PCLK/4, $T_{hold0} = T_{setup1}$ = 0.5 SCK, T_{idle0} = 0.5 SCK, CKP0 = 0 (CSIHnTCSS0) → CKP1 = 1 (CSIHnTCSS1))

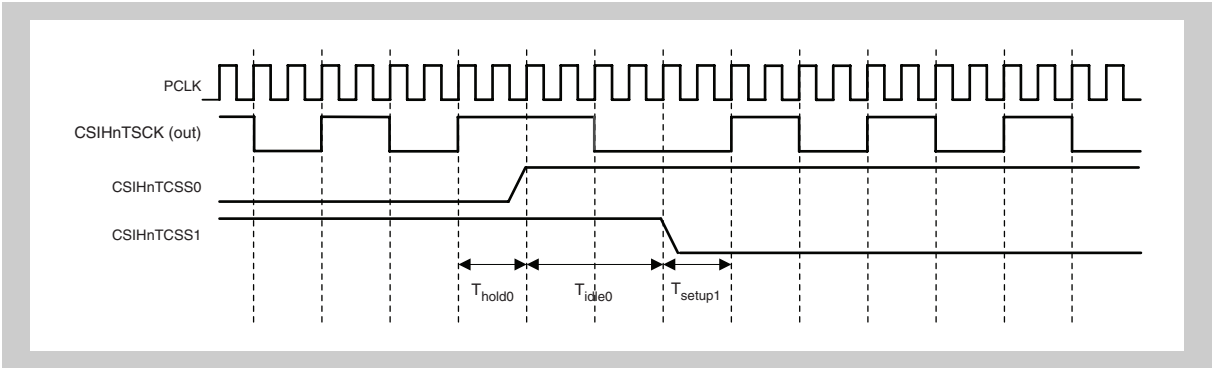


Figure 25-9 Clock phase timing (in the case of PCLK/4, $T_{hold0} = T_{setup1}$ = 0.5 SCK, T_{idle0} = 1.0 SCK, CKP0 = 0 (CSIHnTCSS0) → CKP1 = 1 (CSIHnTCSS1))

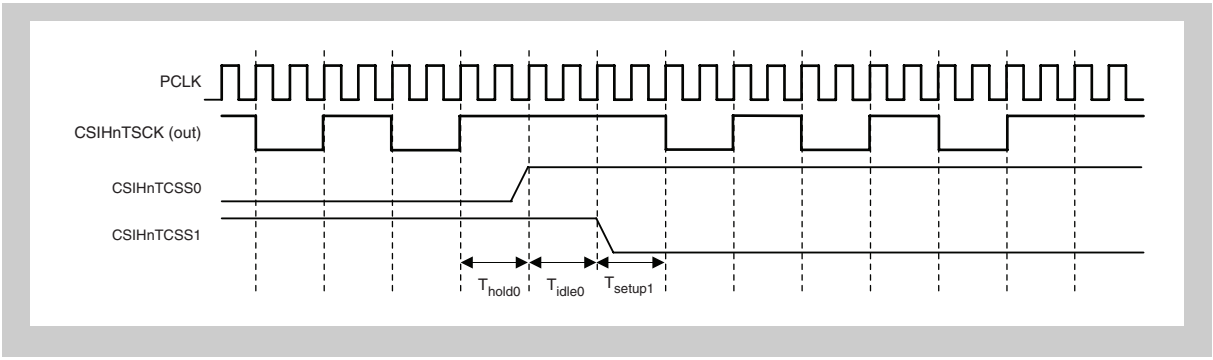


Figure 25-10 Clock phase timing (in the case of PCLK/4, $T_{hold0} = T_{setup1}$ = 0.5 SCK, T_{idle0} = 0.5 SCK, CKP0 = 0 (CSIHnTCSS0) → CKP1 = 0 (CSIHnTCSS1))

(2) Changing the data phase

The bit CSIHnCFGx.CSIHnDAPx defines the phase of the data bits relative to the clock.

If CSIHnCFGx.CSIHnDAPx = 0, the transmission clock CSIHnTSCK holds its level after the last bit of a data is transferred.

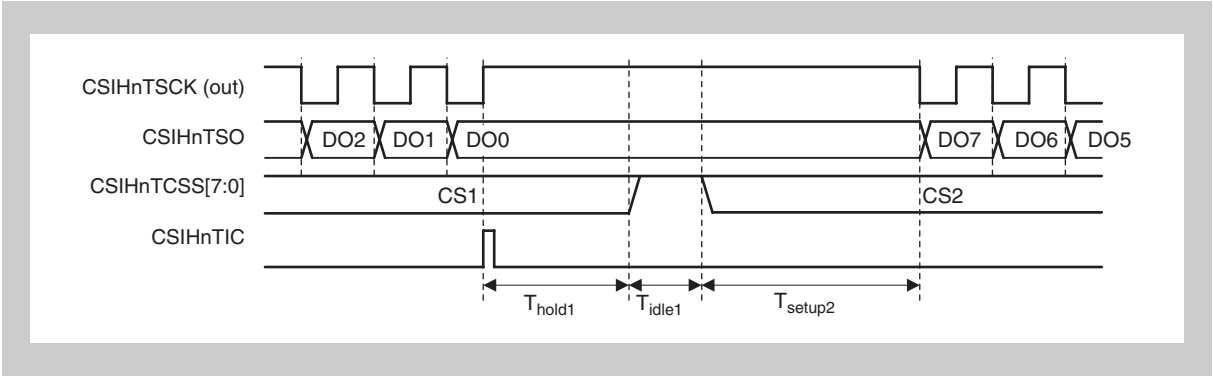


Figure 25-11 Data phase timing with CSIHnCFG1.CSIHnCKP1 = 0, CSIHnCFG1.CSIHnDAP1 = 0 and CSIHnCFG2.CSIHnCKP2 = 0, CSIHnCFG2.CSIHnDAP2 = 0

If the default clock phase changes between two consecutive chip selects, the transmission clock CSIHnTSCK changes its level after the last bit of the first data is transferred:

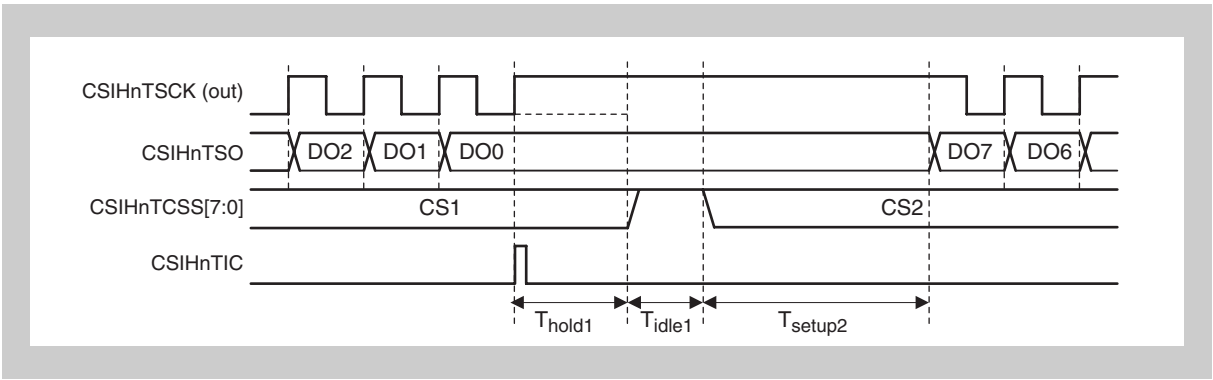


Figure 25-12 Data phase timing with CSIHnCFG1.CSIHnCKP1 = 1, CSIHnCFG1.CSIHnDAP1 = 0 and CSIHnCFG2.CSIHnCKP2 = 0, CSIHnCFG2.CSIHnDAP2 = 1

Note that the minimum idle time of one CSIHnTSCK cycle is automatically inserted, if CSIHnCFGx.CSIHnIDx[2:0] = 0 (T_{idle1} = 0.5 CSIHnTSCK cycles).

25.3.5 The job concept

In terms of CSIH, a job consists of a number of data that are transferred.

Note The job mode can only be used in master mode (CSIHnCTL2.CSIHnPRS[2:0] = 111_B).

Job mode enabled The job mode is enabled and disabled by CSIHnCTL1.CSIHnJE, while the CSIH is disabled by CSIHnCTL0.CSIHnPWR = 0.

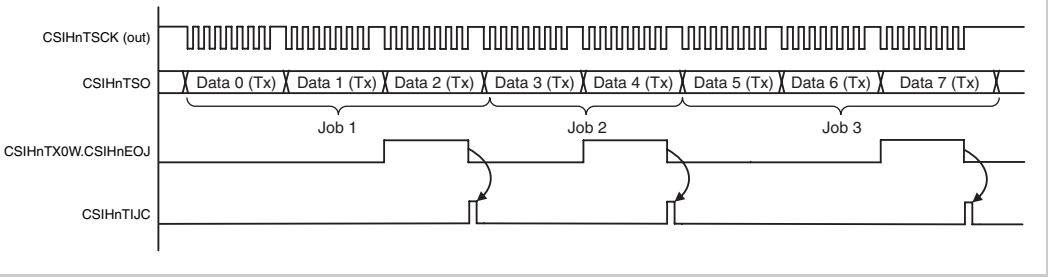


Figure 25-13 Job examples

A job ends when a data with the end-of-job bit set, i.e. with CSIHnTX0W.CSIHnEOJ = 1.

A communication stop can be specified to occur after a job has finished. This is done by setting CSIHnCTL0.CSIHnJOBE. When CSIHnJOBE is set, the communication continues until a data is sent, for which the CSIHnEOJ bit was set. After this data is sent, the communication is stopped and the end-of job-interrupt CSIHnTIJC is generated.

25.3.6 Serial clock selection

In master mode, the transmission baudrate is selectable using

- CSIHnCTL2.CSIHnPRS[2:0]
- CSIHnCTL2.CSIHnBRS[11:0]
- CSIHnCFGx.CSIHn.CSIHnPSCLx

While the settings in the CSIHnCTL2 register determine the transmission base clock CSIHnBPCLK, a chip select dedicated prescaler, controlled by CSIHnCFGx.CSIHn.CSIHnPSCLx, allows to generated different baudrates for different chip selects.

The following figure shows a block diagram of the baudrate generator.

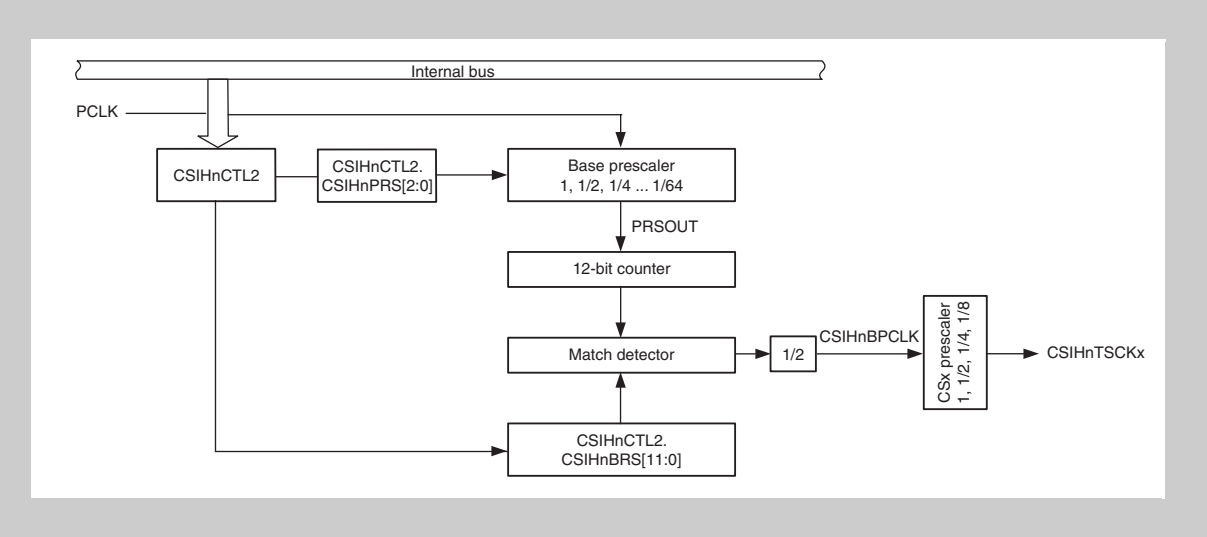


Figure 25-14 Baud rate generator block diagram

Clearing CSIHnCTL2.CSIHnBRS[11:0] disables the baudrate generator, and thus all CSIHnTSCKx are stopped.

Baud rate calculation

The baudrate is calculated as:

$$CSIHnTSCKx = PCLK / (2^m \times k \times 2 \times 2^j)$$

where

m = CSIHnCTL2.CSIHnPRS[2:0] = 0 to 6

k = CSIHnCTL2.CSIHnBRS[11:0] = 1 to 4095

j = CSIHnCFGx.CSIHnPSCLx[1:0]

Baud rate limits

When setting the baud rate, please note:

- Maximum acceptable baud rate in master mode is PCLK / 4.
- Maximum acceptable baud rate in slave mode is PCLK / 6 (must be ensured by the external master).
- Minimum baud rate in both modes is PCLK / 524160.

Caution There might be restrictions on the maximum baud rate that can actually be used depending on the product. Specify the baud rate so as not to exceed the maximum rate for each product.

Example If PCLK = 80 MHz, the maximum baud rate is

- 20.0 Mbps (PCLK / 4) in master mode
- 13.3 Mbps (PCLK / 6) in slave mode

The slowest baud rate is 190.8 bps (PCLK / 524160).

25.3.7 CSIH buffer memory

The CSIH has a configurable RAM that can be used for buffered I/O. The size is 128 words. One word consists of 32 bits of data.

The following configurations are available:

Mode	CSIHnCTL0. CSIHnMBS	CSIHnMCTL0. CSIHnMMS[1:0]
FIFO mode	0	00_B
Dual buffer mode		01_B
Transmit-only buffer mode		10_B
Direct access mode	1	X

(1) FIFO mode

In FIFO mode, data can be written to the CSIHnTX0W register without waiting for completion of the transmission, and data can be received without reading the CSIHnRX0W register immediately. provided the FIFO is not full.

Data to be transmitted is stored to the FIFO memory. Transmission and reception occur simultaneously – one bit is sent, one bit is received. That means, received data overwrites the transmitted data in the FIFO.

The CSIH automatically updates the respective FIFO memory pointers when a data package is committed, sent or received:

Pointer description	Control bits	Range
Number of words that have not been transmitted	CSIHnSTR0.CSIHnSPF[7:0]	0 to 128
Number of words stored in the reception FIFO buffer	CSIHnSTR0.CSIHnSRP[7:0]	0 to 128
Address of data to be sent	CSIHnMRWP0.CSIHnTRWA[6:0]	0000_H to $01FC_H$
Address of received data	CSIHnMRWP0.CSIHnRRA[6:0]	0000_H to $01FC_H$

The CSIH status register contains also two FIFO status flags:

- CSIHnSTR0.CSIHnFLF: FIFO full
- CSIHnSTR0.CSIHnEMF: FIFO empty

When this mode is started, bit CSIHnSTCR0.CSIHnPCT must be set. This resets all FIFO pointers and flags.

(2) Dual buffer mode

In this mode, the memory is divided into two parts of equal size – this means 64 words for transmit data and 64 words for received data. In dual buffer mode, the respective buffer pointers indicate:

Pointer description	Pointers[a]	Range
Destination address for data written to or read from CSIHnTX0W/H	CSIHnMRWP0.CSIHnTRWA[6:0]	0000_H to $00FC_H$
Address of data read from CSIHnRX0W/H	CSIHnMRWP0.CSIHnRRA[6:0]	0000_H to $00FC_H$
Transmission pointer	CSIHnMCTL2.CSIHnSOP[6:0]	0000_H to $00FC_H$

a) Each pointer is automatically incremented after each read or write.

(3) Transmit-only buffer mode

In this mode the entire memory is used to save transmission data.

Received data must be read directly from CSIHnRX0W/H.

In transmit-only buffer mode, the respective buffer pointer is:

Pointer description	Pointer[a]	Range
Destination address for data written to or read from CSIHnTX0W/H	CSIHnMRWP0.CSIHnTRWA[6:0]	0000_H to $01FC_H$
Transmission pointer	CSIHnMCTL2.CSIHnSOP[6:0]	0000_H to $01FC_H$

a) Each pointer is automatically incremented after each read or write.

(4) Direct access mode

In direct access mode, the CSIH memory is completely bypassed:

- Transmission data provided by the CPU to the transmission data register CSIHnTX0W or CSIHnTX0H is directly copied to the shift register.
- Reception data is directly copied from the shift register to the reception data register CSIHnRX0W or CSIHnRX0H.

25.3.8 Data transfer modes

(1) Transmit-only mode

Setting CSIHnCTL0.CSIHnTXE = 1 and CSIHnCTL0.CSIHnRXE = 0 puts the CSIH in transmit-only mode. Start of transmission depends on the memory mode:

- In case of FIFO or direct access mode, transmission starts when transmit data is written to the CSIHnTX0W or CSIHnTX0H register.
- In case of dual buffer or transmit-only buffer mode, transmission starts when bit CSIHnMCTL2.CSIHnBTST is set.

(2) Receive-only mode

Setting CSIHnCTL0.CSIHnTXE = 0 and CSIHnCTL0.CSIHnRXE = 1 puts the CSIH in receive-only mode.

In master mode, the start of reception depends on the memory mode:

- In case of FIFO, transmit-only buffer or direct access mode, reception starts when dummy data is written in the CSIHnTX0W or CSIHnTX0H register.
- In the dual buffer or transmit-only buffer mode, transmission starts when the CSIHnMCTL2.CSIHnBTST bit is set.

In slave mode, reception starts as soon as the transmission clock CSIHnTSCK from the master is received. It is not necessary to write data to the CSIHnTX0W or CSIHnTX0H register of the slave.

(3) Transmit & receive mode

Setting CSIHnCTL0.CSIHnTXE = 1 and CSIHnCTL0.CSIHnRXE = 1 puts the CSIH in transmit / receive mode.

The start of the communication (transmission and reception) depends on the memory mode:

- In case of FIFO or direct access mode, communication starts when transmit data is written to the CSIHnTX0W or CSIHnTX0H register.
- In case of dual buffer or transmit-only buffer mode, communication starts when bit CSIHnMCTL2.CSIHnBTST is set.

(4) Summary

The following table provides a summary. It shows how the data transfer is started in the various memory, operating, and transfer modes.

Table 25-10 Start of data transfer

Memory modes	Transfer modes	Operating modes	Condition for starting a data transfer
FIFO mode, direct access mode	Transmission mode Transmission/reception mode	Master, slave	When transmission data is written to the CSIHnTX0W or CSIHnTX0H register
	Reception mode	Master	When dummy data is written to the CSIHnTX0W or CSIHnTX0H register
		Slave	When the serial clock CSIHnTSCK is received from the master
Transmit-only buffer mode, dual buffer mode	Transmission mode Transmission/reception mode Reception mode	Master, slave	When 1 is written to CSIHnMCTL2.CSIHnBTST

25.3.9 Data length selection

(1) Data length between 7 and 16 bits

CSIHnCFGx.CSIHnDLSx[3:0] can be used to select the data packet length for each chip select signal in the range from seven to 16 bits. The examples below show the communication with MSB first (CSIHnCFGx.CSIHnDIRx = 0).

Data length = 16 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 0000_B):

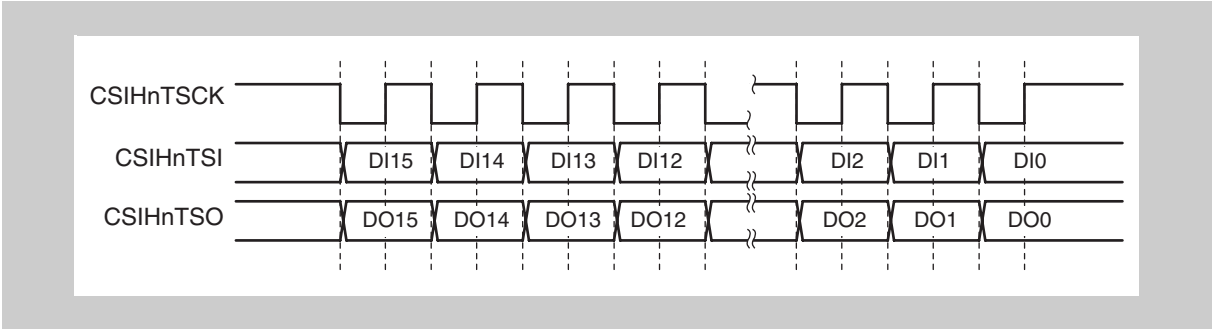


Figure 25-15 16 bit data length, MSB first

Data length = 14 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1110_B):

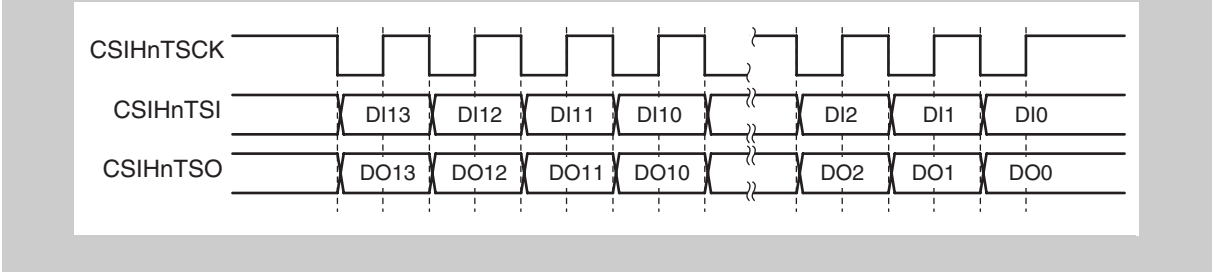


Figure 25-16 14 bit data length, MSB first

(2) Data length greater than 16 bits

If the length of the data to be sent/received exceeds 16 bits, the extended data length (EDL) feature can be used.

The EDL function is enabled by setting CSIHnCTL1.CSIHnEDLE.

The operation and setup procedure of the EDL function are described below:

- Data is divided into 16-bit blocks and a remainder. For example, a 42-bit character string is divided into two 16-bit blocks and a 10-bit remainder.
- For the remainder, the data length is specified for the CSIHnCFGx.CSIHnDLSx[3:0] bits.
- When transmitting 16-bit blocks, set the CSIHnTX0W.CSIHnEDL bit. In this case, the data written to CSIHnTX0W is sent as a 16-bit data length regardless of the CSIHnCFGx.CSIHnDLSx[3:0] bits.
- When the specified length of data (the remainder when CSIHnTX0W.CSIHnEDL = 0) is transmitted, the transfer ends.

Example Example of transmitting the 40-bit data $123456789A_H$ to CS0

The 40-bit data is divided into two 16-bit blocks of data and one 8-bit block of data.

- Initialize CSIHnCFGx.CSIHnDLSx[3:0] = 8_D.
- To send the string $123456789A_H$ with MSB first, write the following sequence to CSIHnTX0W:
  - $2000\ 1234_H$ (CSIHnTX0W.CSIHnEDL = 1)
  - $2000\ 5678_H$ (CSIHnTX0W.CSIHnEDL = 1)
  - $0000\ 009A_H$ (CSIHnTX0W.CSIHnEDL = 0)

The following figure illustrates the timing.

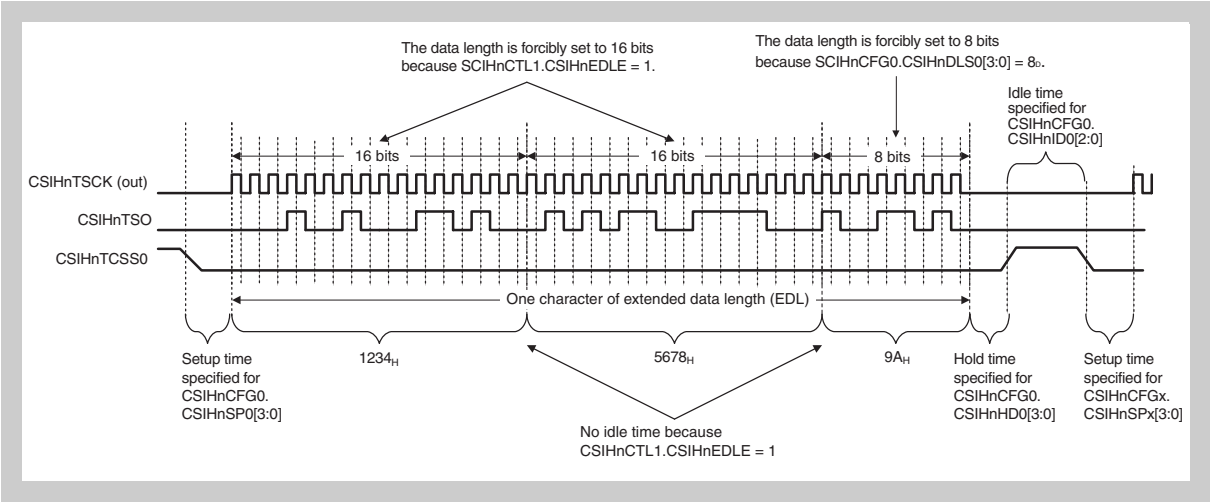


Figure 25-17 EDL timing chart

Notes

1. A data length less than 7 bits can be specified only when the EDL function is used.
2. It is not possible to send two consecutive data with a data length of less than 7 bits.
3. If parity is enabled, the parity bit is added after the last bit.
4. The following describes an example where the transmitted data is 123456_H.
   - CSIHnCFGx.CSIHnDIR is cleared to 0 (MSB first).
     $2000\ 1234_H$ is written to CSIHnTX0W (CSIHnTX0W.CSIHnEDL = 1).
     $0000\ 0056_H$ is written to CSIHnTX0W (CSIHnTX0W.CSIHnEDL = 0).
   - CSIHnCFGx.CSIHnDIR is set to 1 (LSB first).
     $2000\ 3456_H$ is written to CSIHnTX0W (CSIHnTX0W.CSIHnEDL = 1).
     $0000\ 0012_H$ is written to CSIHnTX0W (CSIHnTX0W.CSIHnEDL = 0).
5. The EDL function cannot be used in the slave mode (CSIHnCTL1.CSIHnPRS[2:0] = 1, 1, 1) and reception mode (CSIHnCTL0.CSIHnTXE = 0, CSIHnCTL0.CSIHnRXE = 1).

25.3.10 Serial data direction selection

The serial data direction is selectable for each chip select signal using the CSIHnDIRx bit in the CSIHnCFGx register.

The examples below show the communication for a data length of 8 bit (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B):

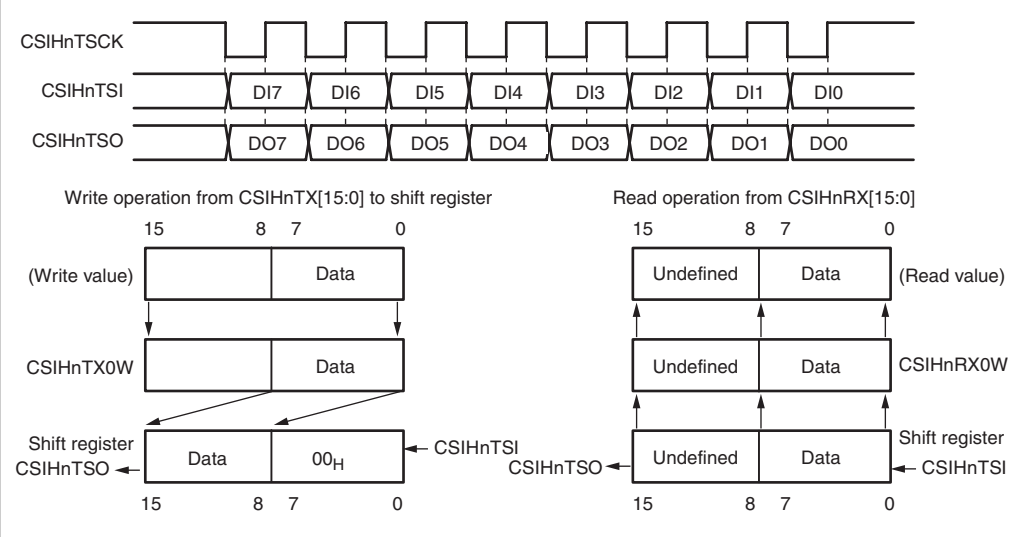


Figure 25-18 Serial data direction select function - MSB first (CSIHnDIR = 0)

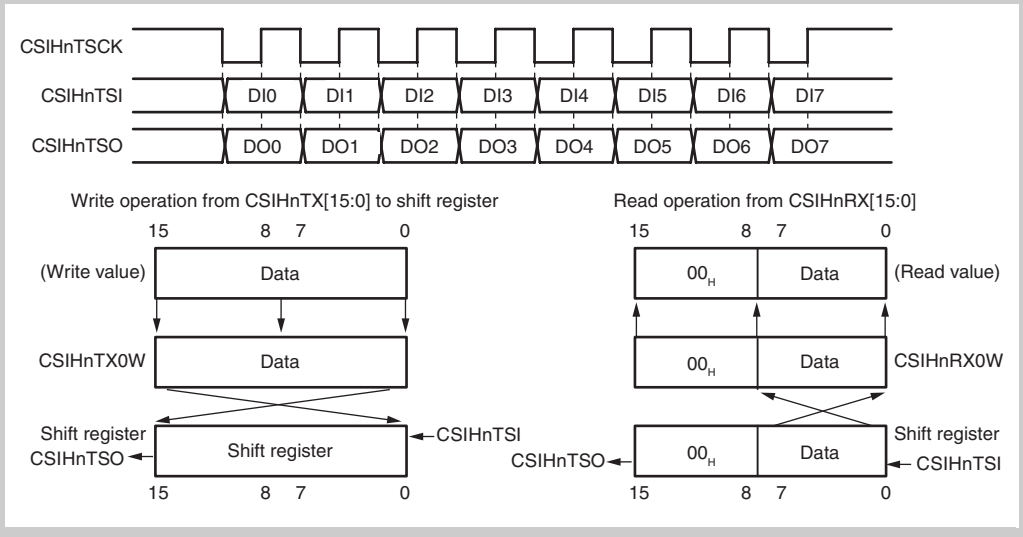


Figure 25-19 Serial data direction select function - LSB first (CSIHnDIR = 1)

25.3.11 Communication in slave mode

The following figure illustrates the communication signals and timings in slave mode.

In slave mode, the data transfer configuration is determined by the CSIHnCFG0 register.

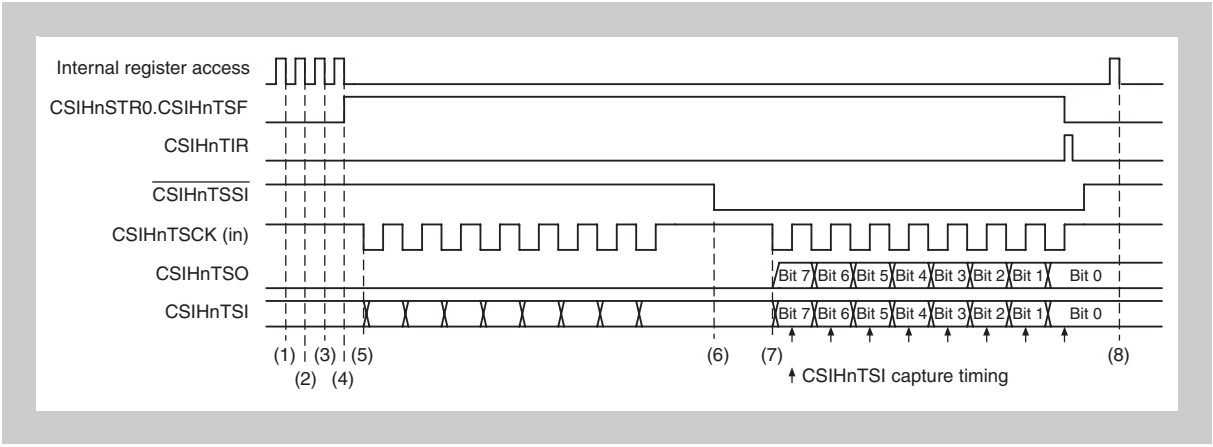


Figure 25-20 Transmit / receive communication timing in slave mode

1. CSIH is placed in the slave mode by setting CSIHnCTL2.CSIHnPRS[2:0] to 111_B. The $\overline{\text{CSIHnTSSI}}$ signal is enabled (CSIHnCTL1.CSIHnSSE = 1).
2. CSIHnCTL1.CSIHnCKR and CSIHnCFG0.CSIHnDAP0 are 0, the data length is 8 bits (CSIHnCFG0.CSIHnDLS0[3:0] = 1000_B), and the data direction is MSB first (CSIHnCFG0.CSIHnDIR0 = 0).
3. CSIH is set to the transmission/reception mode (CSIHnCTL0.CSIHnPWR = 1, CSIHnCTL0.CSIHnTXE = 1, and CSIHnCTL0.CSIHnRXE = 1). Communication is enabled to start.
4. If transfer data is written to the transmission data register CSIHnTX0W or CSIHnTX0H, the transfer status flag CSIHnSTR0.CSIHnTSF is automatically set, and the system waits for the $\overline{\text{CSIHnTSSI}}$ signal to become low.
5. Data transmission/reception does not start while the $\overline{\text{CSIHnTSSI}}$ signal is high, even if the external serial clock CSIHnTSCK is input. CSIHnTSO retains the value, and input to CSIHnTSI is ignored.
6. When the $\overline{\text{CSIHnTSSI}}$ signal becomes low, it indicates that CSIHnTSO is enabled and data transmission is possible.
7. If a serial clock is input while $\overline{\text{CSIHnTSSI}}$ is low, the transfer data is transmitted from CSIHnTSO in synchronization with the serial clock, and data is received from CSIHnTSI at the same time.
8. The CSIHnRX0W or CSIHnRX0H register is read.

Note For details about the operating procedure in the slave mode for each operation mode, see *25.5 “Operating Procedures”*.

25.3.12 CSIH interrupt requests

CSIH can generate the following interrupt requests:

- CSIHnTIC (communication interrupt)
- CSIHnTIR (reception interrupt)
- CSIHnTIRE (error interrupt)
- CSIHnTIJC (job completion interrupt)

(1) CSIHnTIC (communication interrupt)

The conditions for generating CSIHnTIC differ depending on the memory mode and whether the job mode is enabled.

The following table gives an overview.

Table 25-11 CSIHnTIC generation

Memory modes	Interrupt source	
	Job mode disabled CSIHnCTL1.CSIHnJE = 0	**Job mode enabled CSIHnCTL1.CSIHnJE = 1**
FIFO	CSIHnTIC is generated immediately before the transmission data in the FIFO buffer disappears to inform the application that new data must be added. CSIHnTIC is generated when the number of transmission data items remaining in the FIFO buffer, CSIHnSTR0.CSIHnSPF[7:0], becomes equal to CSIHnMCTL1.CSIHnFES[6:0].	
	However, CSIHnTIC is not generated if the job is interrupted[a].	–
Transmit-only buffer, dual buffer	CSIHnTIC is generated when communication ends (as specified by the CSIHnMCTL2.CSIHnND[7:0] bits).	CSIHnTIC is generated when data is transmitted while CSIHnTX0W.CSIHnCIRE is “1”. However, when the data and a job interrupt request[a] are transmitted while CSIHnTX0W.CSIHnCIRE is “1”, CSIHnTIJC is generated instead of CSIHnTIC.
Direct access	CSIHnTIC is generated each time a data transfer is performed. However, CSIHnTIC is not generated if the job is interrupted[a].	Except when communication is interrupted, CSIHnTIC is generated each time a data transfer is performed. However, when the data and a job interrupt request[a] are transmitted while CSIHnTX0W.CSIHnCIRE is “1”, CSIHnTIJC is generated instead of CSIHnTIC.

a) Job abortion condition: CSIHnTX0W.CSIHnEOJ = 1 *and* CSIHnCTL0.CSIHnJOBE = 1

CSIHnTIC in direct access mode

The following example shows the CSIHnTIC behavior in direct access mode.

The following example assumes:

- Master mode
- Direct access memory mode
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- Data length 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

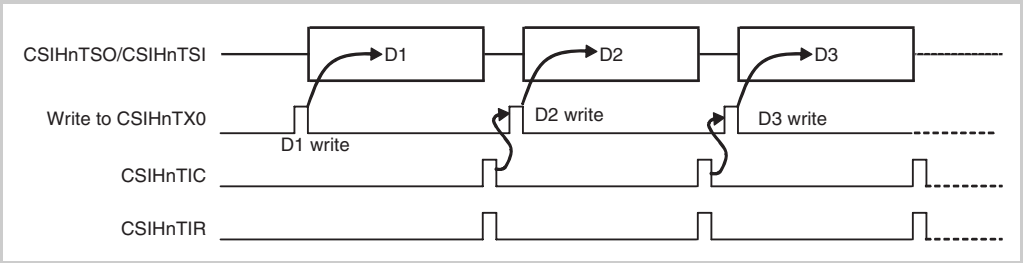


Figure 25-21 Generation of CSIHnTIC after transfer (CSIHnCTL1.CSIHnSLIT = 0)

If job mode is enabled (CSIHnCTL1.CSIHnJE = 1) and a job ends because data is sent with CSIHnTX0W.CSIHnEOJ = 1 and communication stop is requested (CSIHnCTL0.CSIHnJOBE = 1), then CSIHnTIC is replaced by the job completion interrupt CSIHnTIJC.

CSIHnTIC can also be set up to occur as soon as the CSIHnTX0 register is free for the next data. This is specified by setting CSIHnCTL1.CSIHnSLIT = 1.

Note This mode allows faster data transfer but is only available in direct access memory mode.

The effect is illustrated in the figure below.

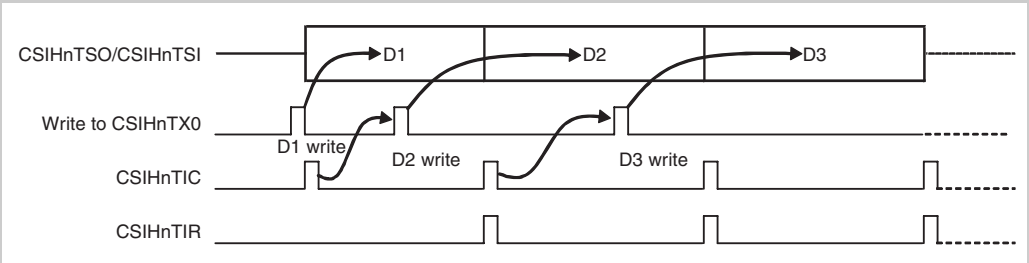


Figure 25-22 Immediate generation of CSIHnTIC (CSIHnCTL1.CSIHnSLIT = 1)

Thus, the new data can be written in advance.

CSIHnTIC in FIFO mode

The following example shows the CSIHnTIC behavior in FIFO mode.

The following example assumes:

- Master mode
- FIFO memory mode
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- Data length 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)

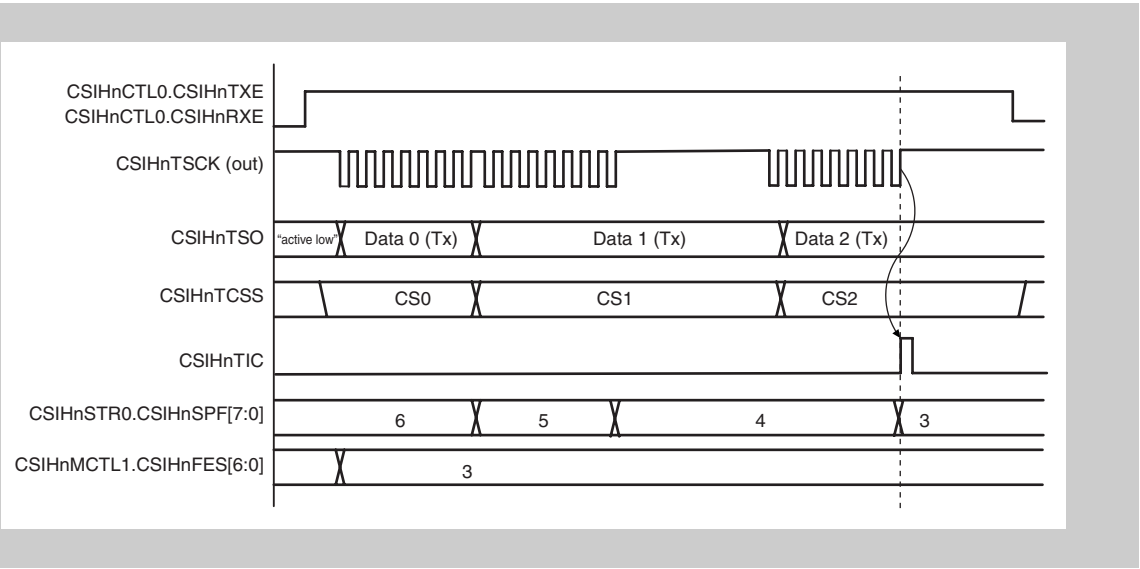


Figure 25-23 Generation of CSIHnTIC in FIFO memory mode

The condition for generating CSIHnTIC in the FIFO mode (an empty reception buffer) is specified for CSIHnMCTL1.CSIHnFES[6:0]. For the example in the above figure, three data items are specified as the condition. The CSIHnSTR0.CSIHnSPF[7:0] bits indicate the number of data items that remain in the FIFO buffer and have not been transmitted. When the number of remaining items matches the condition, the interrupt CSIHnTIC is generated.

CSIHnTIC in job mode

The following example shows the CSIHnTIC behavior in job mode.

The following example assumes:

- Master mode
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- Data length 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

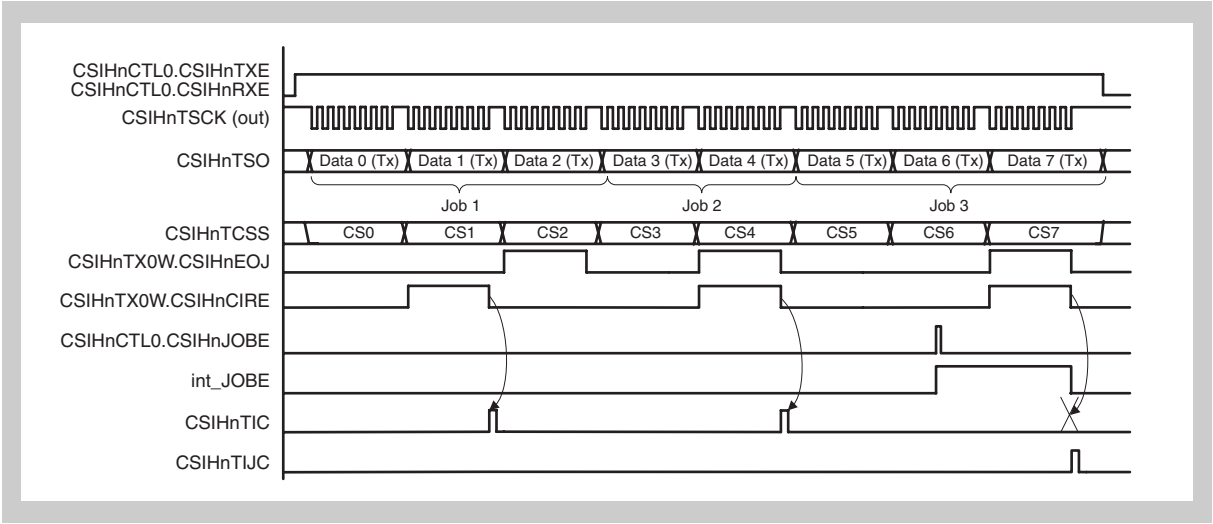


Figure 25-24 Generation of CSIHnTIC in job mode

Note The int_JOBE signal in the above timing chart is the internal signal of the CSIHnJOBE bit.

The rules for generating CSIHnTIC in job mode are:

Table 25-12 Generation of CSIHnTIC in job mode (1/2)

Memory modes	CSIHnTX0W. CSIHnEOJ	CSIHnTX0W. CSIHnCIRE	CSIHnTIC
FIFO mode	0	0	Generated
	0	1	Generated
	1	0	CSIHnCTL0.CSIHnJOBE = 0: Generated
			CSIHnCTL0.CSIHnJOBE = 1: CSIHnTIJC is generated instead of CSIHnTIC.
	1	1	CSIHnCTL0.CSIHnJOBE = 0: Generated
			CSIHnCTL0.CSIHnJOBE = 1: CSIHnTIJC is generated instead of CSIHnTIC.

Table 25-12 Generation of CSIHnTIC in job mode (2/2)

Memory modes	CSIHnTX0W. CSIHnEOJ	CSIHnTX0W. CSIHnCIRE	CSIHnTIC
Dual buffer mode, transmit-only buffer mode	0	0	Not generated
	0	1	Generated
	1	0	CSIHnCTL0.CSIHnJOBE = 0: Generated
			CSIHnCTL0.CSIHnJOBE = 1: CSIHnTIJC is generated instead of CSIHnTIC.
	1	1	CSIHnCTL0.CSIHnJOBE = 0: Generated
			CSIHnCTL0.CSIHnJOBE = 1: CSIHnTIJC is generated instead of CSIHnTIC.
Direct access mode	0	-	Generated
	1	-	CSIHnCTL0.CSIHnJOBE = 0: Generated
			CSIHnCTL0.CSIHnJOBE = 1: CSIHnTIJC is generated instead of CSIHnTIC.

(2) CSIHnTIR reception interrupt

Depending on the memory mode and the job mode, this interrupt is generated according to the following conditions:

Table 25-13 CSIHnTIR interrupt generation

Memory mode	Master and slave	
	Job mode disabled CSIHnCTL1.CSIHnJE = 0	Job mode enabled CSIHnCTL1.CSIHnJE = 1
FIFO	This interrupt occurs when the FIFO buffer is almost full with received data, indicating to the application that the FIFO must be emptied. CSIHnTIR is generated, if the number of received data in the FIFO CSIHnSTR0.CSIHnSRP[7:0] equals CSIHnMCTL1.CSIHnFFS[6:0].	
Dual buffer	The interrupt is generated when communication ends (as specified by the CSIHnMCTL2.CSIHnND[7:0] bits) and CSIHnCTL0.CSIHnRXE = 1.	The interrupt is generated each time data is received if CSIHnCTL0.CSIHnRXE = 1.
Transmit-only buffer, direct access	The interrupt is generated each time data is received if CSIHnCTL0.CSIHnRXE = 1.	

In transmit-only or dual buffer mode, this interrupt is generated in receive-only and transmit/receive mode after each data has been received.

CSIHnTIR in direct access mode

The following example shows the CSIHnTIR behavior in direct access mode.

The following example assumes:

- Master mode
- Direct access mode
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- Data length 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)

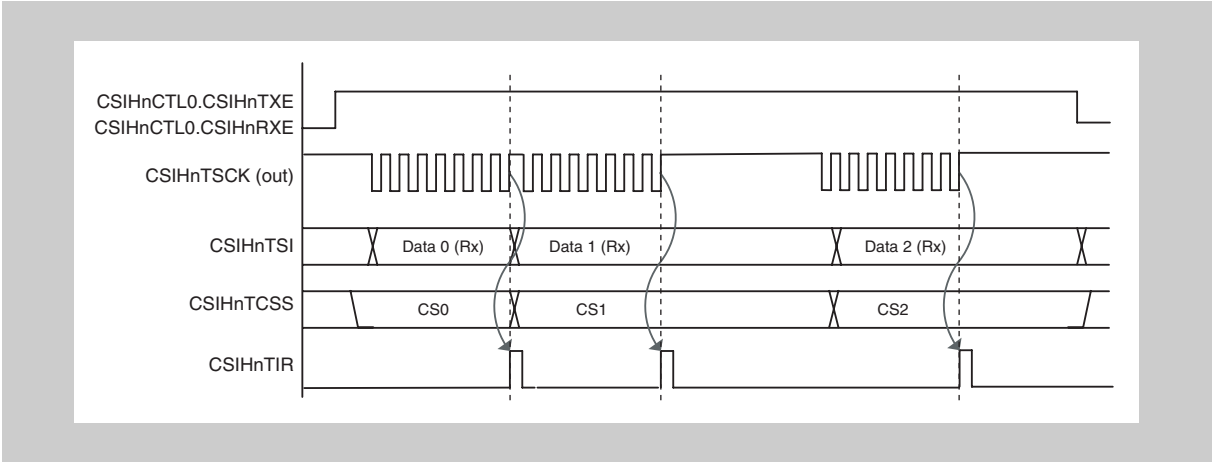


Figure 25-25 Generation of CSIHnTIR in direct access memory mode

CSIHnTIR in the dual buffer mode

The following example shows the CSIHnTIR behavior in buffer mode.

The following example assumes:

- Master mode
- Transmit-only or dual buffer mode
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Default clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- 8-bit data length (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Three data items transmitted (CSIHnMCTL2.CSIHnND[7:0] = 03_H)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)

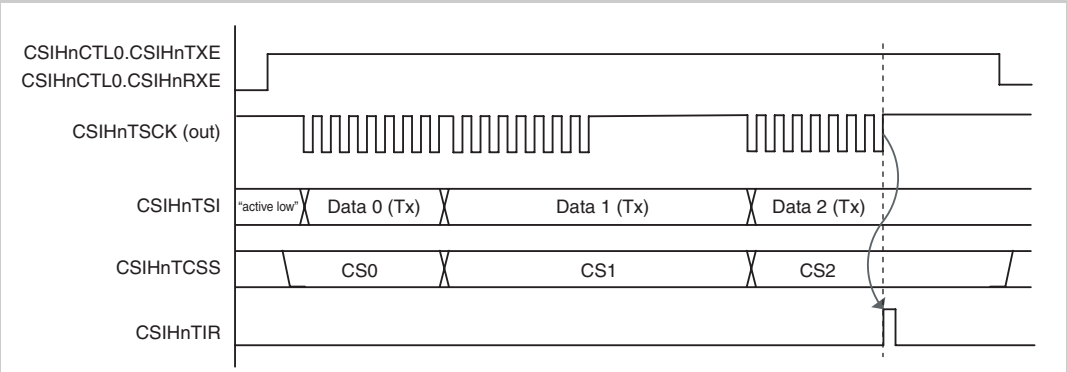


Figure 25-26 CSIHnTIR generation in the dual buffer mode

(3) CSIHnTIRE reception error interrupt

This interrupt is generated whenever an error is detected.

Table 25-14 Data error types

Error type	Communication status after error interrupt	Comment
FIFO overflow error	Interrupt is generated and communication continues	The data written to the FIFO is lost, but previously started communications are continued.
Parity error	Interrupt is generated and communication continues	–
Data consistency error	Interrupt is generated and communication continues	–
Timeout error	Interrupt is generated and communication continues	–
Overrun error	If CSIHnCTL1.CSIHnHSE = 0 (no handshake), communication continues after the interrupt is generated. (Communication does not stop.)	This error occurs (but only for the FIFO mode) if the CPU reads reception data after the number of reception data items reaches 0.
	If CSIHnCTL1.CSIHnHSE = 1 (handshake exists), communication is stopped according to the handshake. No interrupt is generated, and no overrun error occurs.	This error occurs when data reception finishes in one of the following statuses: • FIFO mode: The FIFO is full. (Overwriting is performed but the pointer is not incremented.) • Transmit-only buffer mode or direct access mode: Reception data remains in the CSIHnRX0 register.

The type of error that caused the generation of CSIHnTIRE is flagged in register CSIHnSTR0.

Additionally a parity and data consistency error flag is attached to the reception data in CSIHnRX0W.

For details about the various error types, see *25.3.14 “Error detection” on page 1874*.

(4) CSIHnTIJC job completion interrupt

This interrupt supports the handling of jobs – see *25.3.4 "Chip select timing details" on page 1848*. This interrupt is only available in master mode.

Job mode is enabled by setting CSIHnCTL1.CSIHnJE = 1. When CSIHnCTL1.CSIHnJE = 0, CSIHnTIJC is not generated.

Depending on the memory mode, this interrupt is generated according to the following conditions:

Table 25-15 CSIHnTIJC interrupt generation

Memory mode	Interrupt source	
	Job mode disabled CSIHnCTL1.CSIHnJE = 0	Job mode enabled CSIHnCTL1.CSIHnJE = 1
FIFO mode	Not applicable	After job abortion[a] is triggered, communication stops on job completion.
Transmit-only buffer mode		
Dual buffer mode		
Direct access mode		

a) Job abortion condition: CSIHnTX0W.CSIHnEOJ = 1 and CSIHnCTL0.CSIHnJOBE = 1

(5) Delay for all interrupts

In the master mode, all interrupts generated by the master can be delayed one half cycle of the serial clock CSIHnTSCK. This function cannot be used in the slave mode.

To specify this delay, set the CSIHnCTL1.CSIHnSIT bit to 1.

The figure below shows an example of using the interrupt delay function with the following settings: CSIHnCTL1.CSIHnSIT = 1 (interrupt delay enabled), CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0 (normal clock phase and data phase), and CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B (8-bit data length).

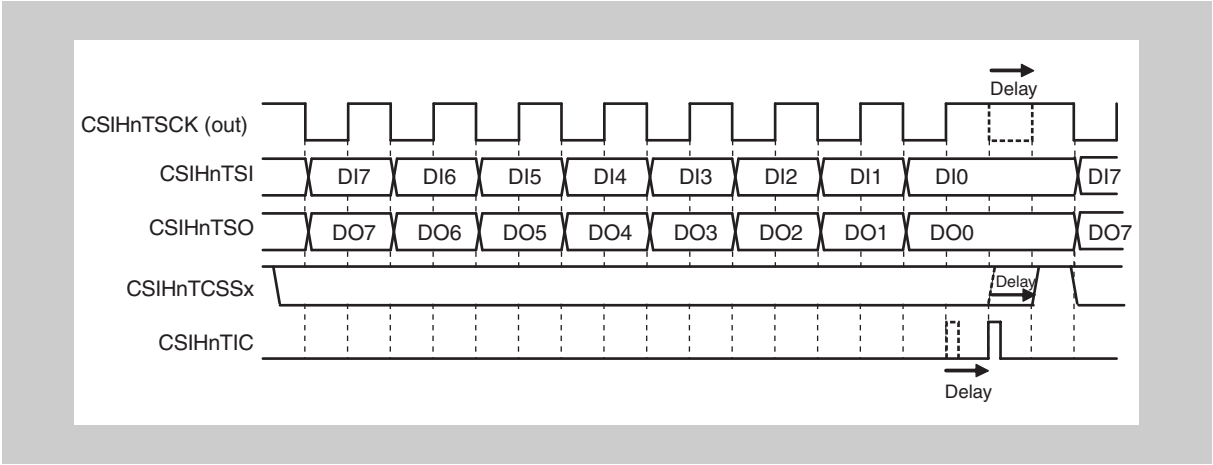


Figure 25-27 Interrupt delay function (CSIHnCTL1.CSIHnSIT = 1)

When CSIHnCTL1.CSIHnSIT is set to 1, a delay of half a serial clock cycle is added. This also delays the end of the current chip select signal (CSIHnTCSSx).

25.3.13 Handshake function

CSIH features a handshake function to synchronize the master and the slave devices. This function can be enabled/disabled by bit CSIHnCTL1.CSIHnHSE. For handshake, the CSIHnTRY signal is used. In addition, when the handshake function is disabled (CSIHnCTL1.CSIHnHSE = 0), the CSIHnTRY signal is output at the low level.

The timing depends on the data phase selection bit CSIHnCFGx.CSIHnDAPx.

(1) Slave mode

When CSIHnCTL1.CSIHnHSE = 1 and the slave is busy, the CSIHnTRY signal is output at the low level. The busy state refers to the following two cases:

- When the slave is not ready for transmission
- When the transmission data cannot be transferred from the shift register to the CSIHnRXW/H register because received data remains in the CSIHnRXW/H register

Also, the state differs according to the memory mode. *Table 25-16 "When the slave is not read for transmission"* and *Table 25-17 "When the transmission data cannot be transferred from the shift register"* describe the state for each memory mode.

Table 25-16 When the slave is not read for transmission

Memory mode	State
	CSIHnCTL0.CSIHnRXE = 1
FIFO mode	The next transmission data buffer is empty. (CSIHnSTR0.CSIHnEMF = 1)
Direct access mode	The next transmission data buffer is empty.
Dual buffer mode	CSIHnMCTL2.CSIHnBTST is not set to 1.
Transmit-only buffer mode	

Table 25-17 When the transmission data cannot be transferred from the shift register

Memory mode	State
	CSIHnCTL0.CSIHnRXE = 1
FIFO mode	The FIFO buffer is full. (CSIHnSTR0.CSIHnFLF = 1)
Direct access mode	The CSIHnRX0W/H register is full.
Dual buffer mode	–
Transmit-only buffer mode	The CSIHnRX0W/H register is full.

Note When the master mode is specified and the CSIHnTRY (in) signal is detected to be at the low level, the following transmission is put on hold and the system waits. While the system waits, the CSIHnTRY (in) signal level is checked every one-half clock cycle of CSIHnTCSS0. However, the chip select signal CSIHnTCSSx is output before the system starts waiting, and is held during the wait.

Cautions

1. For the master, specify settings so that only the CSIHnTRY pin is detected from the slave selected by CSIHnTCSSx.
2. Even if an active level signal from the slave to the master is detected during transmission, the next transmission is not performed until the current transmission operation ends.

Two descriptions are provided below, one for the FIFO mode and one for the direct access mode.

When the memory mode is FIFO mode

The slave is in transmit-only or transmit/receive mode but has no transmission data in its buffer. This status is indicated by the flag CSIHnSTR0.CSIHnEMF.

The following examples assume a data length of 8 bits.

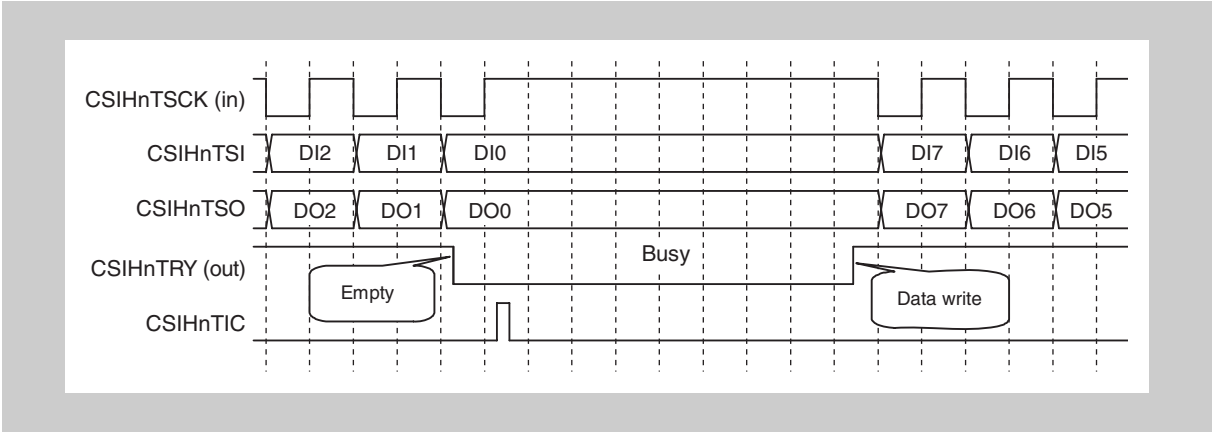


Figure 25-28 Busy signal from the slave (FIFO mode, CSIHnCFGx.CSIHnDAPx = 0)

The slave sets CSIHnTRY to high (“ready”) as soon as new transmission data is written to the FIFO.

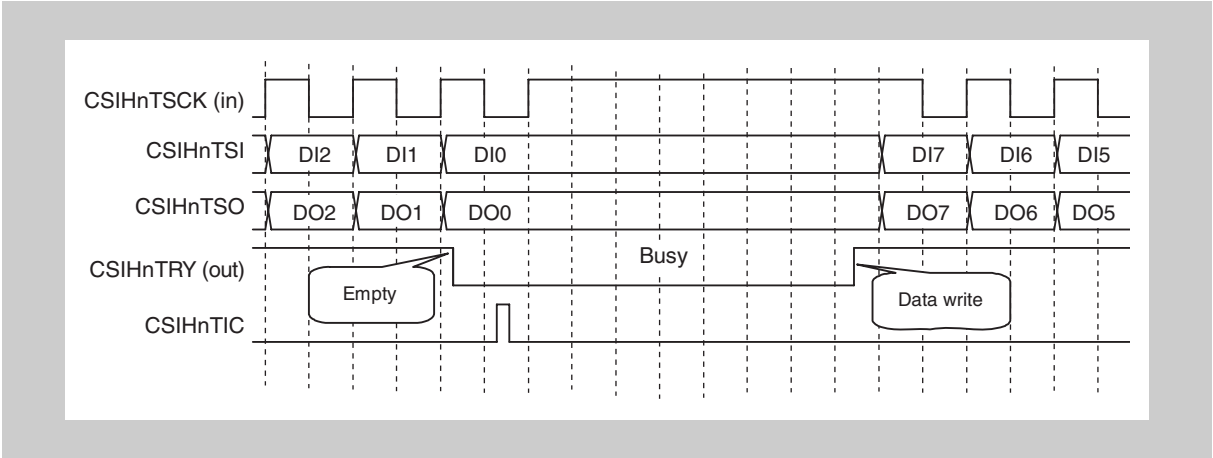


Figure 25-29 Busy signal from the slave (FIFO mode, CSIHnCFGx.CSIHnDAPx = 1)

When the memory mode is direct access mode

The slave is in receive-only or transmit/receive mode but previously received data is still in the CSIHnRX0 register, and new data cannot be copied from the shift register to CSIHnRX0 (CSIHnRX0 full condition).

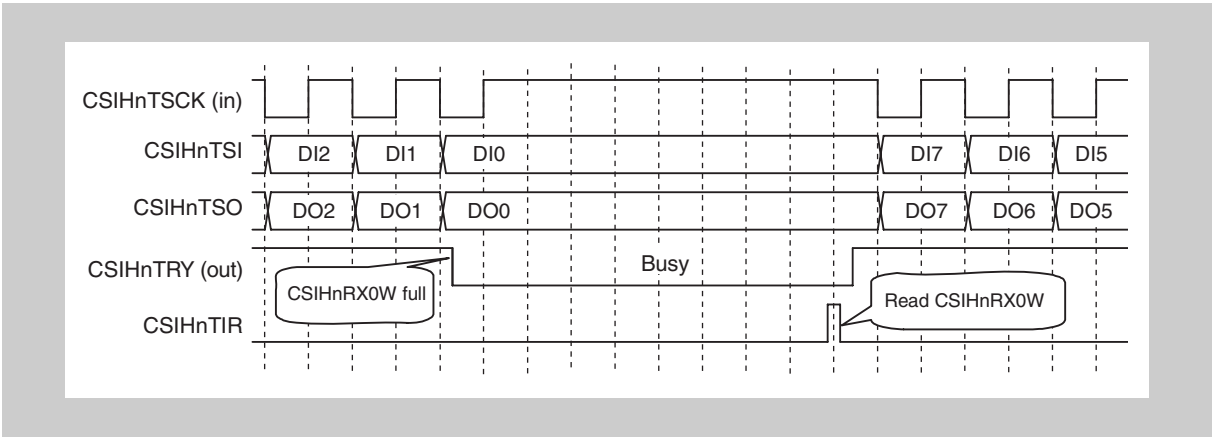


Figure 25-30 Busy signal from the slave (direct access mode, CSIHnCFGx.CSIHnDAPx = 0)

The slave sets CSIHnTRY to high (“ready”) as soon as the reception data register CSIHnRX0 has been read.

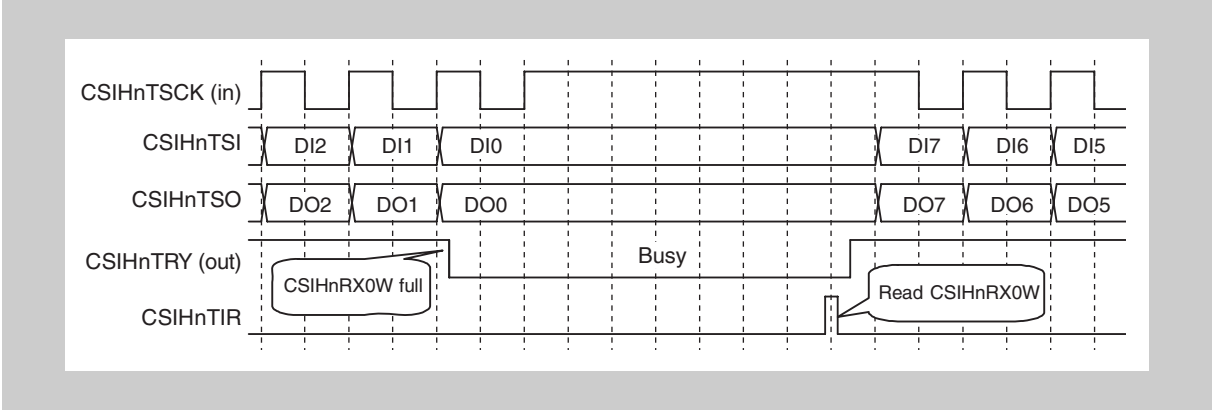


Figure 25-31 Busy signal from the slave (direct access mode, CSIHnCFGx.CSIHnDAPx = 1)

(2) Master mode

When the master detects that CSIHnTRY is at the low level, the following transfer is put on hold, and the master goes on stand by. It suspends the clock CSIHnTSCK.

The CSIHnTRY level is checked at each half clock cycle of CSIHnTSCK.

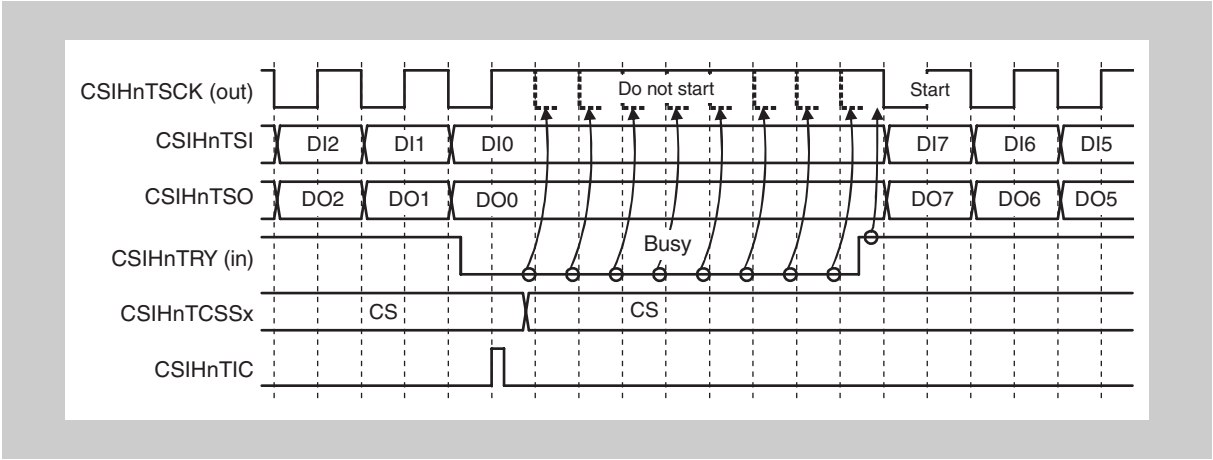


Figure 25-32 Master's reaction on CSIHnTRY (CSIHnCFGx.CSIHnDAPx = 0)

If the CSIHnTRY signal is pulled down while a data transfer is in progress, the serial clock is suspended after the transfer is complete.

The master resumes the communication as soon as CSIHnTRY becomes high (slave is "ready").

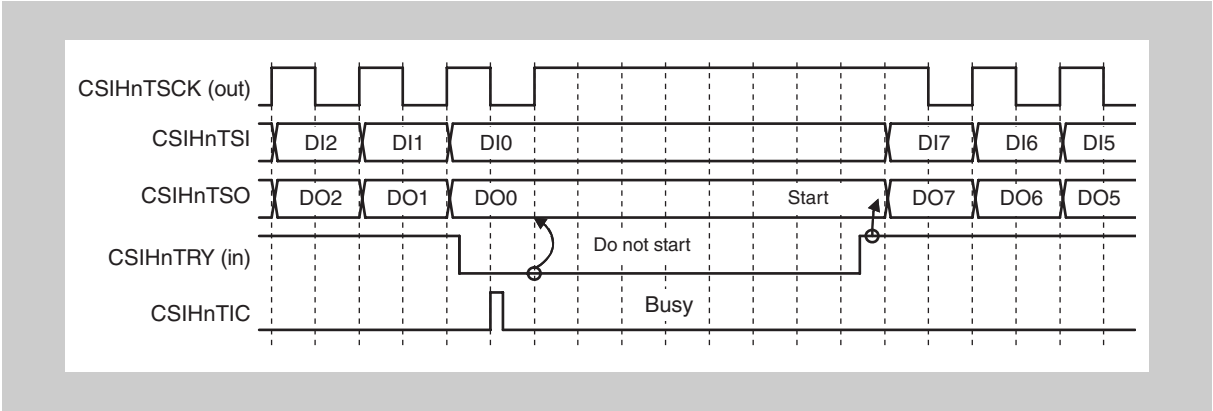


Figure 25-33 Master's reaction on CSIHnTRY (CSIHnCFGx.CSIHnDAPx = 1)

Caution If multiple slaves are connected, the master must only detect the CSIHnTRY signal of the slave it has selected for communication.

The slave must set the CSIHnTRY signal to low before the next data transfer starts. If the CSIHnTRY signal is set to low during a data transfer, the data transfer continues to the end.

25.3.14 Error detection

The CSIH can detect five error types:

- Data consistency error (transmission data)
- Parity error (received data)
- Overrun error (received data)
- Timeout error (in FIFO mode)
- Overflow error (in FIFO mode)

Check for parity, data consistency and timeout errors can be enabled/disabled individually.

If one of these errors is detected, the interrupt request CSIHnTIRE is generated and the corresponding flag is set.

(1) Data consistency check

The purpose of the data consistency check is to ensure that the data physically sent as output signal is identical with the original data that was copied to the shift register.

The data consistency check can be enabled/disabled by bit CSIHnCTL1.CSIHnDCS. It is not active if data transmission is disabled (CSIHnCTL0.CSIHnTXE = 0).

When the data consistency check is active, the data transferred from CSIHnTX0W or CSIHnTX0H to the shift register is copied to a separate register. In addition, the physical levels at CSIHnTSO are read back via the CSIHnTDCS signal into an own shift register.

After completion of the transmission, the sent data is compared with the original transmission data.

Mismatch is considered as a data consistency error.

When a data consistency error occurs:

- Interrupt CSIHnTIRE is generated.
- Bit CSIHnSTR0.CSIHnDCE is set.

Additionally, CISHnRX0W.CSIHnTDCE is set with the corresponding data.

The function is illustrated in the following block diagram.

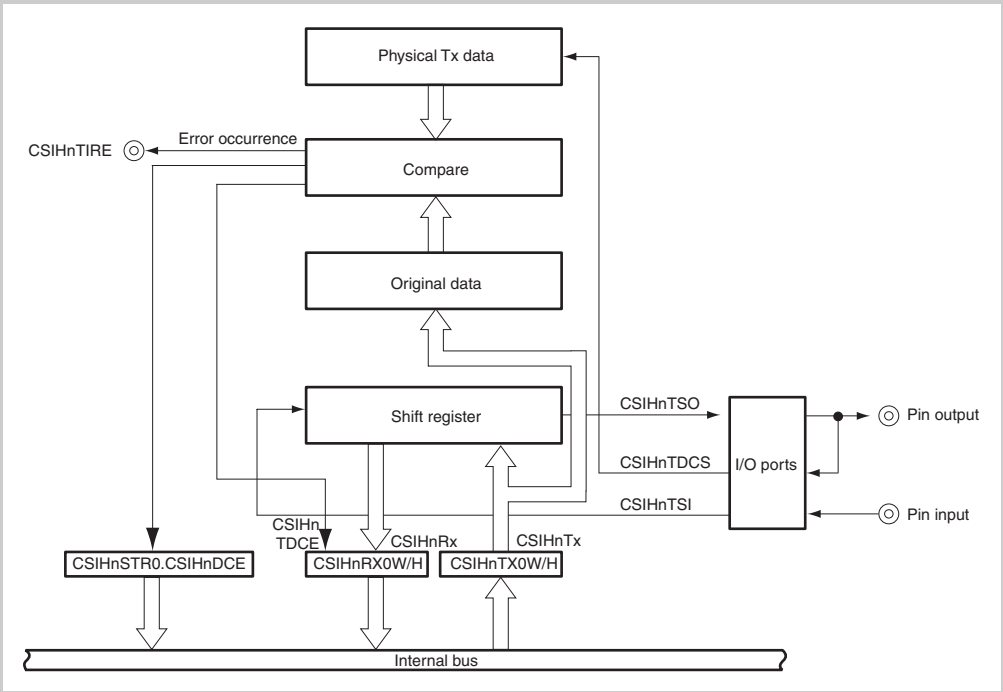


Figure 25-34 Data consistency check functional block diagram

(2) Parity check

Parity checks are often used to detect single bit errors during data transmission. CSIH can append a parity bit to the last data bit (even if extended data length is used).

The use and type of parity is specified in CSIHnCFGx.CSIHnPSx[1:0].

Parity check is enabled if CSIHnCFGx.CSIHnPSx[1] = 1.

The parity bit is checked after a reception is complete.

When a parity error occurs:

- Interrupt CSIHnTIRE is generated.
- Bit CSIHnSTR0.CSIHnPE is set.

Additionally, CISHnRX0W.CSIHnRPE is set with the corresponding data.

The following figure shows an example.

- Data length is 8 bits.
- The data transmitted is 05_H and 35_H.
- Data direction is LSB first.
- Parity type is odd.

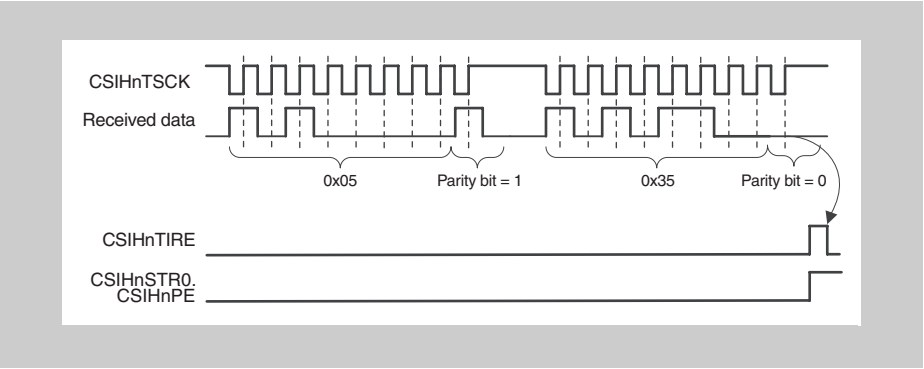


Figure 25-35 Parity check example

The parity bit of the first data is 1. There is no parity error, because the total number of ones (including the parity bit) is odd.

The parity bit of the second data is 0. This is detected as a parity error, because the total number of ones (including the parity bit) is even.

If using the extended data length (EDL) function, the parity bit is added after the last data bit.

(3) Timeout error

Timeout error checks are only possible in slave FIFO mode.

A timeout error occurs if neither of the following occurs within a specific time:

- Reading reception data in the FIFO buffer
- Reception of data by the FIFO buffer from CSIHnTSI

The time is defined in CSIHnMCTL0.CSIHnTO[4:0] in multiples of 8 times the transmission clock CSIHnSCK. Timeout error occurs when the specified time is exceeded (When CSIHnMCTL0.CSIHnTO[4:0] is cleared to 00000_B, the timeout time is not detected.).

A dedicated timeout counter measures the time between the last and the next read operation.

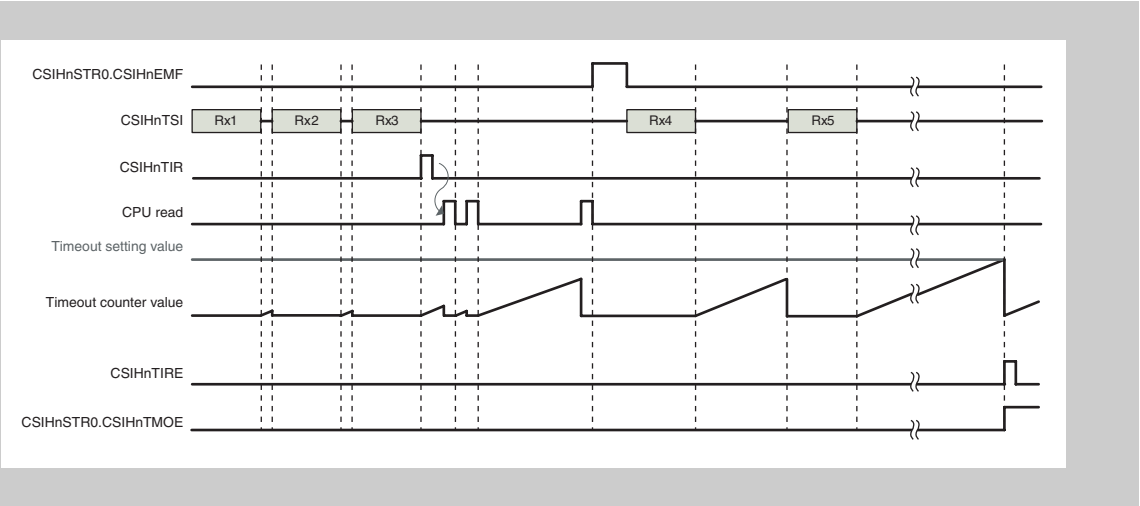


Figure 25-36 Timeout error check functional timing chart

The timeout counter starts when:

- Reception ends.
- Reading data from the CPU ends. (If the buffer is empty, the counter does not start.)
- A timeout error is detected.

After a timeout error is detected, if the system is left as is, the timeout counter restarts.

If the time specified for the CSIHnMCTL0.CSIHnTO[4:0] bits is reached again, another CSIHnTIR interrupt is output.

The timeout counter continues counting as long as reception data is not read. To stop the timeout counter, read all the reception data, or set CSIHnSTCR0.CSIHnPCT (to 1). However, the pointer is cleared in this case.

The timeout counter is reset when:

- Data is read.
- One new data item is received.
- A timeout error is detected.
- The CSIHnSTCR0.CSIHnPCT bit is set.

When a timeout error occurs:

- Interrupt CSIHnTIRE is generated.
- Bit CSIHnSTR0.CSIHnTMOE is set.

(4) Overflow error

Overflow errors can occur in the FIFO mode. An overflow error occurs when transmission data is written to the CSIHnTX0W or CSIHnTX0H register while the FIFO buffer is full of transmission data and reception data.

Example 100 data have been transmitted. That means, the FIFO contains 100 received data. The application starts to read the received data.

While the read operation is in progress, the application begins to write another set of 50 transmission data to the FIFO. However, only 10 received data have been read up to now, 90 are still in the FIFO.

In this case, only 38 cells are available for new transmission data. When the CPU tries to write the 39th data, an overflow error happens.

This is illustrated in the following figure:

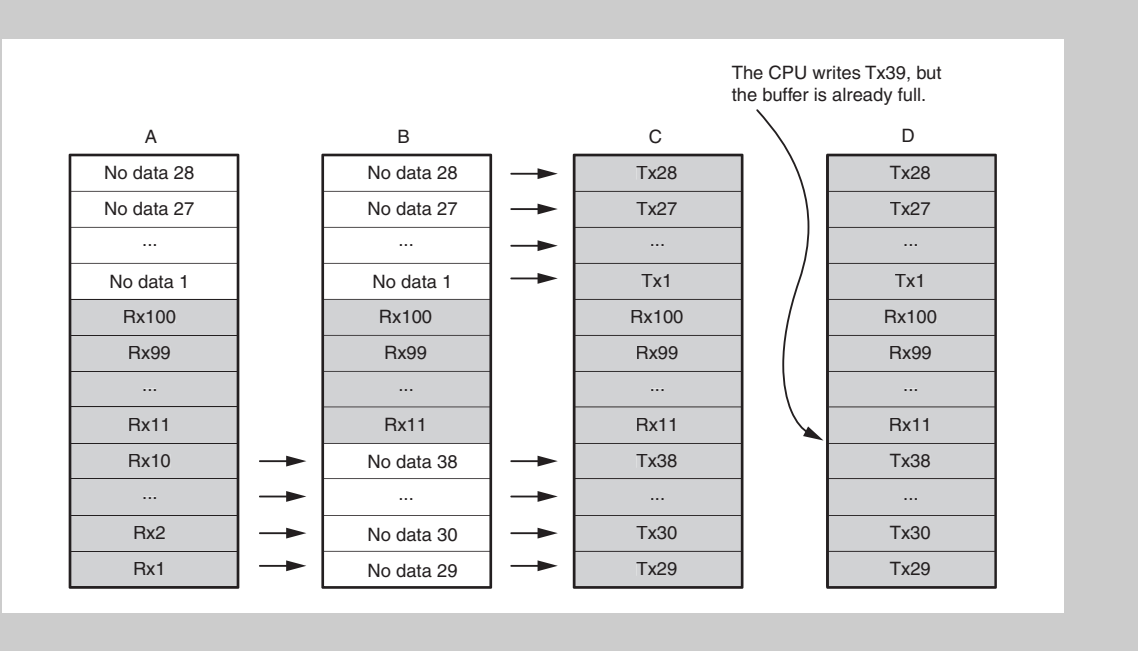


Figure 25-37 FIFO overflow

The data after 39 are discarded. The following figure shows the associated timing.

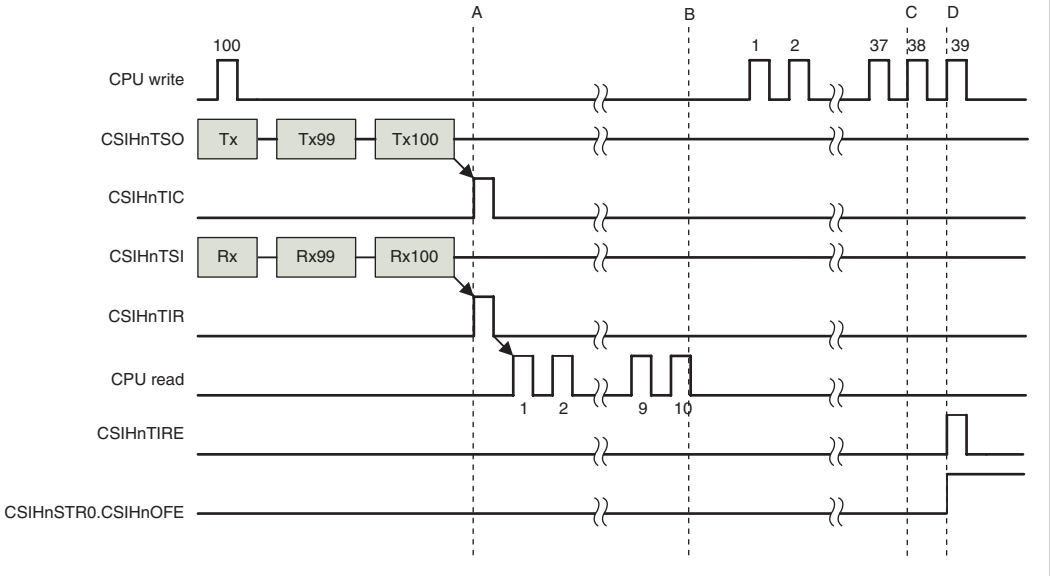


Figure 25-38 FIFO overflow timing

When an overflow error occurs:

- Interrupt CSIHnTIRE is generated.
- Bit CSIHnSTR0.CSIHnOFE is set.

(5) Overrun error

Overrun errors can occur in the direct access mode, transmit-only buffer mode, and FIFO mode. They cannot occur in the dual buffer mode.

Direct access / transmit-only buffer

In direct access and transmit-only buffer mode, this error occurs when newly received data cannot be transferred from the shift register to the reception data register CSIHnRX0. This happens when CSIHnRX0 was not read and therefore contains previous reception data.

In the master mode, because the serial clock is stopped until the CPU reads reception data, overrun errors do not occur. In the slave mode, the handshake function can be used to avoid this problem.

The following figure illustrates the function.

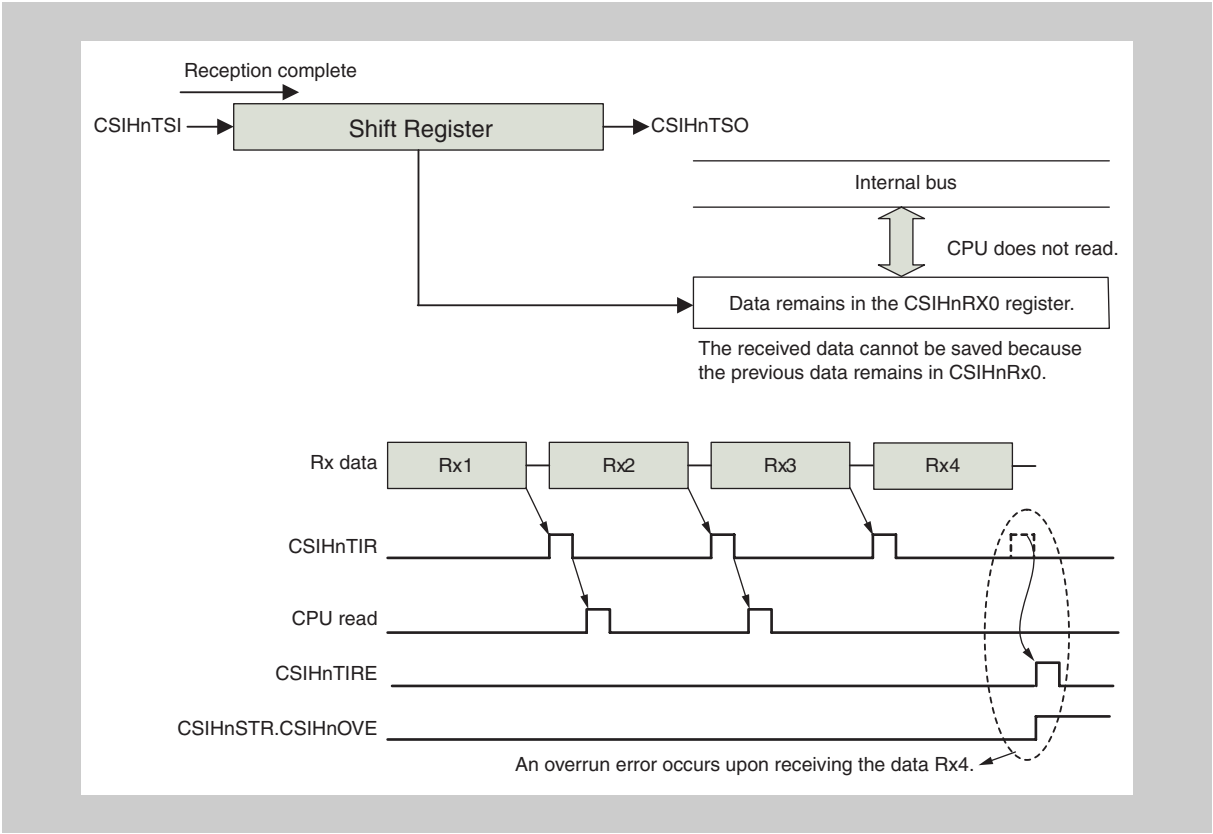


Figure 25-39 Overrun error detection in direct access and transmit-only buffer mode

FIFO mode In FIFO mode, an overrun error occurs if:

1. FIFO buffer full
   Because the FIFO buffer is full, new received data cannot be transferred from the shift register to the FIFO buffer.
2. No data
   The CPU attempts to read reception data that does not exist.

Note If the CPU attempts to read reception data that does not exist in the FIFO mode, an overrun error occurs even if data reception is disabled (CSIHnCTL0.CSIHnRXE = 0).

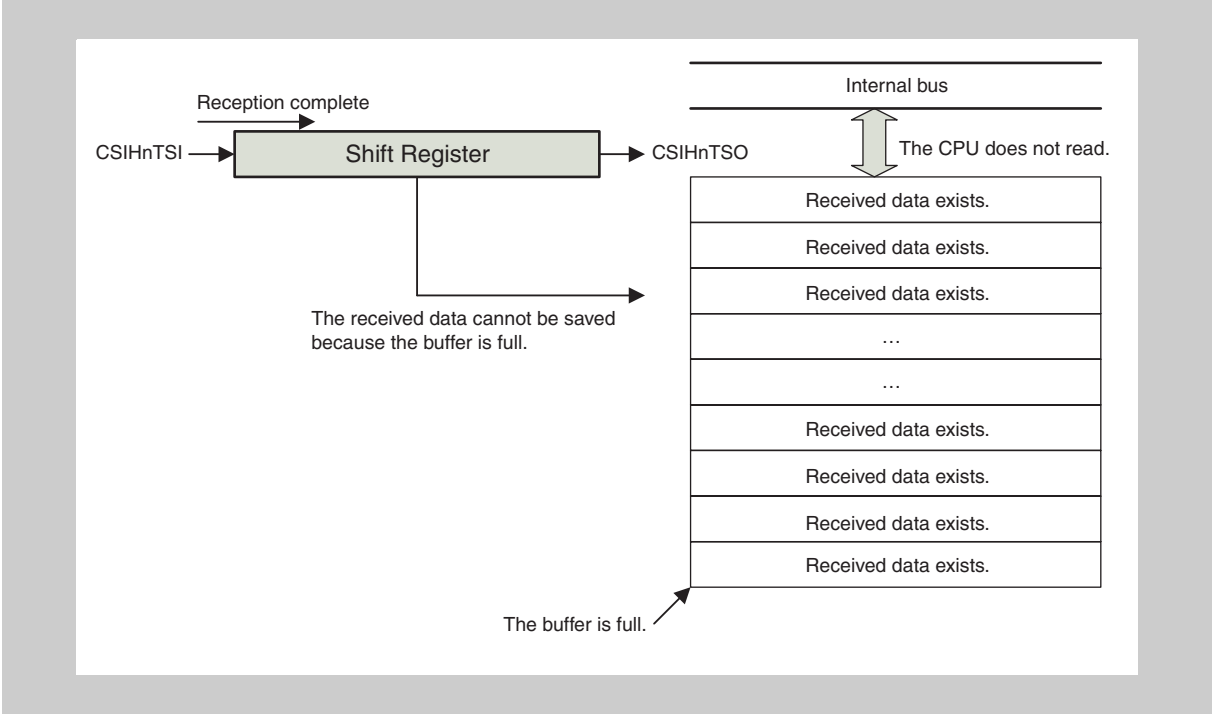


Figure 25-40 Overrun error detection in FIFO mode (FIFO full)

3. The CPU attempts to read non existing reception data

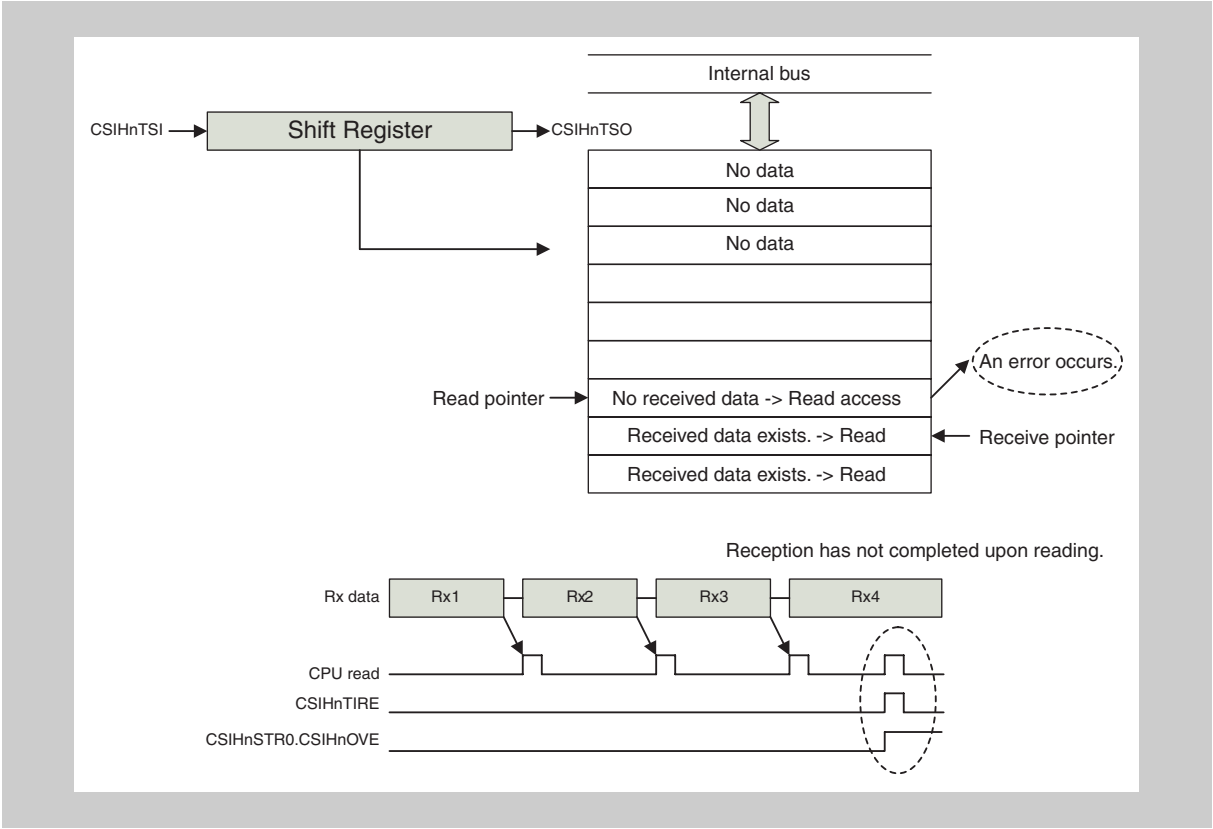


Figure 25-41 Overrun error detection in FIFO mode (no data)

When an overrun error occurs:

- Interrupt CSIHnTIRE is generated.
- Bit CSIHnSTR0.CSIHnOVE is set.
- Communication is stopped (except if the CPU tried to read non existing data).
- Communication continues (unless the CPU attempted to read data that did not exist).

Note An overrun error can be avoided in slave mode by using the handshake. When handshake is used in slave mode, the receiver (slave) signals to the transmitter (master) that it is busy. The transmitter then waits until the receiver has read its reception data register and is ready again.

For details, see *25.3.13 “Handshake function” on page 1870*.

25.3.15 Loop-back mode

Loop-back mode is a special mode for self-test. This feature is only available in master mode.

When this mode is active, the transmit and receive signals are internally connected, as shown in the figures below. The signals CSIHnTSCK, CSIHnTSO, and CSIHnTSI are disconnected from the ports. In addition, the CSIHnTSO output level is fixed to low, and CSIHnTSCK becomes inactive according to the setting of CSIHnCFGx.CSIHnCKPx. The handshake function cannot be used at this time. The rest of CSIH works as in normal operation.

The CSIHnTSCK, CSIHnTSO, CSIHnTSI, and CSIHnTCSSn[7:0] signals are disconnected from ports. The CSIHnTSO signal is fixed to the low output level, and the CSIHnTSCK and CSIHnTCSSn[7:0] signals are set to the inactive level (the level specified for the CSIHnCFGx.CSIHnCKPx bit in the case of the CSIHnTSCK signal, and the level specified for the CSIHnCTL1.CSIHnCLS[7:0] bits in the case of the CSIHnTCSSn[7:0] signal).

To perform a self-test of the CSIH, CSIHnCTL1.CSIHnLBM is set to 1, and a normal transfer operation is executed. Next, whether the reception data and transmission data are the same is checked.

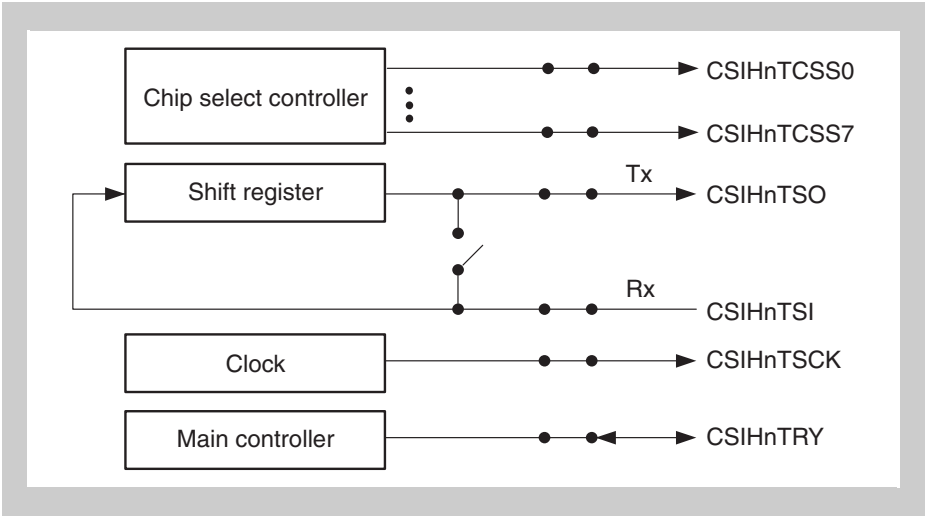


Figure 25-42 Normal operation

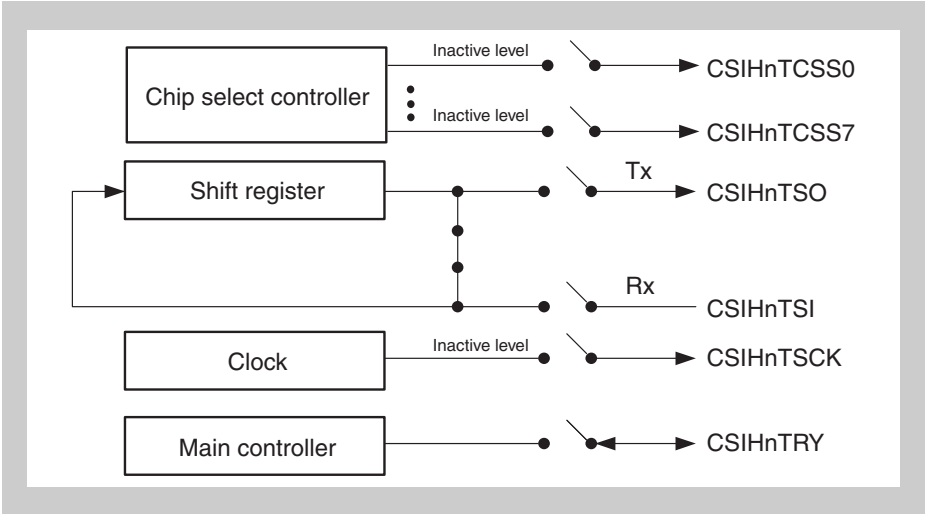


Figure 25-43 Loop-back operation

25.4 CSIH Control Registers

CSIHn is controlled and operated by the following registers:

Table 25-18 CSIH register overview

Register name	Symbol	Address
Control register 0	CSIHnCTL0	<CSIHn_base> + 0000_H
Control register 1	CSIHnCTL1	<CSIHn_base> + 0010_H
Control register 2	CSIHnCTL2	<CSIHn_base> + 0014_H
Status register 0	CSIHnSTR0	<CSIHn_base> + 0004_H
Status clear register 0	CSIHnSTCR0	<CSIHn_base> + 0008_H
Memory control register 0	CSIHnMCTL0	<CSIHn_base> + 1040_H
Memory control register 1	CSIHnMCTL1	<CSIHn_base> + 1000_H
Memory control register 2	CSIHnMCTL2	<CSIHn_base> + 1004_H
Transmit data register 0 for word access	CSIHnTX0W	<CSIHn_base> + 1008_H
Transmit data register 0 for half word access	CSIHnTX0H	<CSIHn_base> + $100C_H$
Receive data register 0 for word access	CSIHnRX0W	<CSIHn_base> + 1010_H
Receive data register 0 for half word access	CSIHnRX0H	<CSIHn_base> + 1014_H
Memory read/write pointer register 0	CSIHnMRWP0	<CSIHn_base> + 1018_H
Configuration register 0	CSIHnCFG0	<CSIHn_base> + 1044_H
Configuration register 1	CSIHnCFG1	<CSIHn_base> + 1048_H
Configuration register 2	CSIHnCFG2	<CSIHn_base> + $104C_H$
Configuration register 3	CSIHnCFG3	<CSIHn_base> + 1050_H
Configuration register 4	CSIHnCFG4	<CSIHn_base> + 1054_H
Configuration register 5	CSIHnCFG5	<CSIHn_base> + 1058_H
Configuration register 6	CSIHnCFG6	<CSIHn_base> + $105C_H$
Configuration register 7	CSIHnCFG7	<CSIHn_base> + 1060_H

<CSIHn_base> The base addresses <CSIHn_base> of the CSIHn is defined in the first section of this chapter under the key word “Register addresses”.

25.4.1 CSIH register details

(1) CSIHnCTL0 - CSIHn control register 0

This register controls CSIHn. It mainly enables or disables the operation clock, transmission/reception, and the memory assigned to transmission/reception. It forces the stop of communication at the end of the current job.

Access This register can be read or written in 8-bit or 1-bit units.

Address <CSIHn_base> + 0000_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
CSIHn PWR	CSIHn TXE	CSIHn RXE	0	0	0	CSIHn JOBE	CSIHn MBS
R/W	R/W	R/W	R	R	R	R/W	R/W

Table 25-19 CSIHnCTL0 register contents

Bit position	Bit name	Function
7	CSIHnPWR	Controls the operation clock. 0: Stops operation clock 1: Provides operation clock Clearing CSIHnPWR to 0 resets the internal circuits, stops operation, and sets the CSIH to standby state. No clock is provided to internal circuits, thus the power consumption of the CSIHn is minimized. If CSIHnPWR is cleared during communication, ongoing communication is immediately aborted. In this case, it is necessary to restart communication from the beginning.
6	CSIHnTXE	Enables/disables transmission. 0: Transmission disabled 1: Transmission enabled
5	CSIHnRXE	Enables/disables reception. 0: Receive disabled 1: Receive enabled
1	CSIHnJOBE	Stops the communication at the end of the current job (Communication ends when data is written to the transmission buffer while CSIHnTX0W.CSIHnEOJ = 1 (indicating that the job has ended).). 0: Communication stop is not required 1: Communication stop This bit can be used to abort an ongoing job. It is automatically cleared. Even if this bit is set, 0 is always returned when it is read. In FIFO mode, the next communication should then be started after clearing the pointers by setting CSIHnSTCR0.CSIHnPCT = 1. **Caution:** CSIHnJOBE is only valid when CSIHnCTL1.CSIHnJE = 1. Setting this bit is prohibited in the slave mode. When this bit is read, 0 is always returned.
0	CSIHnMBS	Bypasses the memory for transmission and/or reception data. 0: Memory mode CSIH memory is used for transmission and/or reception data 1: Direct access mode CSIH memory is bypassed **Caution:** In the slave mode, perform rewriting at the same time that CSIHnCTL0.CSIHnPWR changes from 0 to 1.

Cautions

1. When CSIHnPWR = 0, do not change the CSIHnTXE, CSIHnRXE, CSIHnJOBE, or CSIHnMBS bit. However, the CSIHnTXE, CSIHnRXE, or CSIHnMBS bit can be changed at the same time that the CSIHnPWR bit changes from 0 to 1.
2. Do not modify CSIHnTXE or CSIHnRXE or CSIHnMBS while a data transmission is pending or going on, i.e. if CSIHnSTR0.CSIHnTSF = 1.

(2) CSIHnCTL1 - CSIHn control register 1

This register controls CSIHn. It mainly specifies the clock phase, interrupt timing, and interrupt delay mode, controls the extended data length, and enables or disables the data consistency check, loopback mode, handshake function, and job mode. This register also selects the active output level of each chip select signal and the chip select signal operation to perform after the last data is transferred.

Access This register can be read or written in 32-bit units.

Address <CSIHn_base> + 0010_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	CSIHn SLIT
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHn CSL7	CSIHn CSL6	CSIHn CSL5	CSIHn CSL4	CSIHn CSL3	CSIHn CSL2	CSIHn CSL1	CSIHn CSL0	CSIHn EDLE	CSIHn JE	CSIHn DCS	CSIHn CSRI	CSIHn LBM	CSIHn SIT	CSIHn HSE	CSIHn SSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIHnCTL0.CSIHnPWR = 0.

Table 25-20 CSIHnCTL1 register contents (1/2)

Bit position	Bit name	Function
17	CSIHnCKR	Selects the CSIHnTSCK clock phase. 0: The default CSIHnTSCK level is the high level. 1: The default CSIHnTSCK level is the low level. **Caution:** When using this bit without using the chip select function, clear CSIHnCFGx.CSIHnCKPx to 0.
16	CSIHnSLIT	Selects the timing of interrupt CSIHnTIC. 0: Normal interrupt timing (interrupt is generated after the transfer) 1: When the contents of the CSIHnTX0W or CSIHnTX0H register are transferred to the shift register, an interrupt is immediately generated. (This only functions in the direct access mode.) For details, see *"CSIHnTIC in direct access mode" on page 1862*.
15 to 7	CSIHnCSLx	Selects the active output level of chip select signal x (CSIHnTCSSx). 0: Chip select is active low 1: Chip select is active high For details, see *25.3.3 "Chip selection (CS) features" on page 1846*.
7	CSIHnEDLE	Enables/disables extended data length (EDL) mode. 0: Extended data length mode disabled 1: Extended data length mode enabled For details, see *(2) "Data length greater than 16 bits" on page 1857*.

Table 25-20 CSIHnCTL1 register contents (2/2)

Bit position	Bit name	Function
6	CSIHnJE	Enables/disables job mode. 0: Job mode disabled 1: Job mode enabled For details, see *25.3.4 "Chip select timing details" on page 1848*. The CSIHnCTL0.CSIHnJOBE, CSIHnTX0W.CSIHnEOJ, and CSIHnTX0W.CSIHnCIRE bits are only valid when this bit is 1. Setting this bit is prohibited in the slave mode.
5	CSIHnDCS	Enables/disables data consistency check. 0: Data consistency check disabled 1: Data consistency check enabled For details, see *(1) "Data consistency check" on page 1874*.
4	CSIHnCSRI	Defines chip select behavior after last data transfer. 0: Chip select holds active level 1: Chip select returns to inactive level The last data is identified at the interrupt timing while in the direct access mode or FIFO mode. The direct access mode is used while CSIHnCTL1.CSIHnSLIT = 1.
3	CSIHnLBM	Controls loop-back mode (LBM). 0: Loop-back mode deactivated 1: Loop-back mode activated For details, see *25.3.15 "Loop-back mode" on page 1884*. Setting this bit is prohibited in the slave mode.
2	CSIHnSIT	Selects interrupt delay mode. 0: No delay 1: Half clock delay for all interrupts This bit is only valid in master mode. In slave mode, no delay is generated. For details, see *"CSIHnTIC in direct access mode" on page 1862*.
1	CSIHnHSE	Enables/disables handshake mode. 0: Handshake function disabled 1: Handshake function enabled For details, see *25.3.13 "Handshake function" on page 1870*.
0	CSIHnSSE	Enables/disables slave select function. 0: Input signal $\overline{\text{CSIHnTSSI}}$ is ignored 1: Input signal $\overline{\text{CSIHnTSSI}}$ is recognized If the slave select function is not used, this bit must be cleared to 0 (see *25.3.2 "Master/slave connections" on page 1844*).

Details about CSIHnCTL1.CSIHnSSE:

Table 25-21 Operation of the slave select function during reception

CSIHnCTL0. CSIHnRXE	CSIHnCTL1. CSIHnSSE	$\overline{\text{CSIHnTSSI}}$	Receive operation
0	–	–	Reception disabled
1	0	–	Possible
1	1	0	Possible
1	1	1	Disabled

Table 25-22 Operation of the slave select function during transmission

CSIHnCTL0. CSIHnTXE	CSIHnCTL1. CSIHnSSE	$\overline{\text{CSIHnTSSI}}$	Transmit operation
0	–	–	Transmission disabled
1	0	–	Possible
1	1	0	Possible
1	1	1	Disabled

(3) CSIHnCTL2 - CSIHn control register 2

This register controls CSIHn. It selects the operating mode, prescaler, and baud rate.

For details, see *25.3.6 "Serial clock selection" on page 1851*.

Access This register can be read or written in 16-bit units.

Address <CSIHn_base> + 0014_H

Initial Value $E000_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnPRS[2:0]			0	CSIHnBRS[11:0]											
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIHnCTL0.CSIHnPWR = 0.

Table 25-23 CSIHnCTL2 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15 to 13</td><td>CSIHnPRS [2:0]</td><td>Selects the operating mode and the value of the prescaler.
<table>
<tr><th>CSIHn PRS2</th><th>CSIHn PRS1</th><th>CSIHn PRS0</th><th>Base clock (PRSOUT) selection</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>PCLK (master mode)</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>PCLK/2 (master mode)</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>PCLK/4 (master mode)</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>PCLK/8 (master mode)</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>PCLK/16 (master mode)</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>PCLK/32 (master mode)</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>PCLK/64 (master mode)</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>External clock by way of CSIHnTSCK (in) (slave mode)</td></tr>
</table></td></tr>
<tr><td>11 to 0</td><td>CSIHnBRS [11:0]</td><td>Selects the baud rate.
<table>
<tr><th>CSIHnBRS[11:0]</th><th>Baud rate at CSIHnTBLK</th></tr>
<tr><td>0</td><td>BRG is stopped</td></tr>
<tr><td>1</td><td>PCLK/(2^m × 1 × 2)</td></tr>
<tr><td>2</td><td>PCLK/(2^m × 2 × 2)</td></tr>
<tr><td>3</td><td>PCLK/(2^m × 3 × 2)</td></tr>
<tr><td>4</td><td>PCLK/(2^m × 4 × 2)</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>4095</td><td>PCLK/(2^m × 4095 × 2)</td></tr>
</table>
Note:
m = 0 to 6: Value specified for CSIHnPRS[2:0]</td></tr>
</table>

(4) CSIHnSTR0 - CSIHn status register 0

This register indicates the status of the CSIH.

Access This register can be read in 32-bit units.

Address <CSIHn_base> + 0004_H

Initial Value $8000\ 0010_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSIHnSRP[7:0]								CSIHnSPF[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHn TMOE	CSIHn OFE	0	0	0	0	0	0	CSIHn TSF	0	CSIHn FLF	CSIHn EMF	CSIHn DCE	0	CSIHn PE	CSIHn OVE
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 25-24 CSIHnSTR0 register contents (1/3)

Bit position	Bit name	Function
31 to 24	CSIHnSRP[7:0]	Indicates the number of received words in FIFO mode. **CSIHnSRP[7:0]** / **Description** 00_H / Number of received words (0 to 128_D) … 80_H Other than the above / Setting prohibited These bits are cleared by CSIHnSTCR0.CSIHnPCT. In the dual buffer mode or transmit-only buffer mode, because the number of data items is managed according to CSIHnMCTL2.CSIHnND[7:0], these bits are fixed to 00_H. They are also fixed to 00_H in the direct access mode because there is no pointer.
23 to 16	CSIHnSPF[7:0]	Indicates the number of unsent data in FIFO mode. **CSIHnSPF[7:0]** / **Description** 00_H / Number of unsent data (0 to 128_D) … 80_H Other than the above / Setting prohibited These bits are cleared by CSIHnSTCR0.CSIHnPCT. In the dual buffer mode or transmit-only buffer mode, because the number of data items is managed according to CSIHnMCTL2.CSIHnND[7:0], these bits are fixed to 00_H. They are also fixed to 00_H in the direct access mode because there is no pointer.

Table 25-24 CSIHnSTR0 register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15</td><td>CSIHnTMOE</td><td>Timeout error flag for the FIFO mode
Indicates whether a timeout error was detected in the FIFO mode.
0: No timeout error was detected in the FIFO mode.
1: A timeout error was detected in the FIFO mode.
For details, see (3) “Timeout error” on page 1877.
This bit is cleared by CSIHnSTCR0.CSIHnTMOEC.
This bit can be written to when CSIHnSTCR0.CSIHnPWR = 0.
This bit is initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0.
If this bit is set due to a timeout error being detected and cleared by CSIHnSTCR0.CSIHnTMOEC at the same time, setting the bit is prioritized.</td></tr>
<tr><td>14</td><td>CSIHnOFE</td><td>Overflow error flag for the FIFO mode
Indicates whether an overflow error was detected in the FIFO mode.
0: No overflow error was detected in the FIFO mode.
1: An overflow error was detected in the FIFO mode.
For details, see (4) “Overflow error” on page 1879.
This bit is cleared by CSIHnSTCR0.CSIHnOFEC.
This bit can be written to when CSIHnSTCR0.CSIHnPWR = 0.
This bit is initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0.
If 129 transmission data items are written to the CSIHnTX0W or CSIHnTX0H register when CSIHnCTL0.CSIHnPWR = 0, an overflow error occurs.
If this bit is set due to an overflow error being detected and cleared by CSIHnSTCR0.CSIHnOFEC at the same time, setting the bit is prioritized.</td></tr>
<tr><td>7</td><td>CSIHnTSF</td><td>Transfer status flag
0: Idle state
1: Transmission is in progress or being prepared
Setting and clearing of this bit is as follows:
<table>
<tr><th rowspan="2">Master mode</th><th colspan="2">Set condition</th><th rowspan="2">Clear condition</th></tr>
<tr><th>Direct access mode, FIFO mode</th><th>Dual buffer mode, transmit-only buffer mode</th></tr>
<tr><td>Transmission mode</td><td rowspan="3">Writing to transmit data register</td><td rowspan="3">Setting CSIHnMCTL2.CSIHnBTST</td><td rowspan="3">Within 0.5 clock cycles from the last CSIHnTSCK edge</td></tr>
<tr><td>Transmission/reception mode</td></tr>
<tr><td>Reception mode</td></tr>
</table>
<table>
<tr><th rowspan="2">Slave mode</th><th colspan="2">Set condition</th><th rowspan="2">Clear condition</th></tr>
<tr><th>Direct access mode, FIFO mode</th><th>Dual buffer mode, transmit-only buffer mode</th></tr>
<tr><td>Transmission mode</td><td rowspan="2">Writing to transmit data register</td><td rowspan="3">Setting CSIHnMCTL2.CSIHnBTST</td><td rowspan="3">Within 0.5 clock cycles from the last CSIHnTSCK edge</td></tr>
<tr><td>Transmission/reception mode</td></tr>
<tr><td>Reception mode</td><td>CSIHnTSCK input timing</td></tr>
</table></td></tr>
</table>

Table 25-24 CSIHnSTR0 register contents (3/3)

Bit position	Bit name	Function
5	CSIHnFLF	Indicates whether the buffer is full while in the FIFO mode. 0: FIFO buffer is not full 1: FIFO buffer is full This bit is set when the sum of the CSIHnSTR0.CSIHnSRP[7:0] bit value and CSIHnSTR0.CSIHnSPF[7:0] bit value matches 80_H, and is cleared when this sum does not match 80_H. This bit is cleared by CSIHnSTCR0.CSIHnPCT. The FIFO buffer sometimes fills up with data that has not been transmitted and reception data.
4	CSIHnEMF	Indicates whether the buffer is empty while in the FIFO mode. 0: FIFO buffer is not empty 1: FIFO buffer is empty This bit is set by SCIHnSTCR0.CSIFnPCT. This bit is set when the sum of the CSIHnSTR0.CSIHnSRP[7:0] bit value and CSIHnSTR0.CSIHnSPF[7:0] bit value matches 00_H, and is cleared when this sum does not match 00_H. The FIFO buffer sometimes does not contain any data that has not been transmitted or reception data.
3	CSIHnDCE	Data consistency error flag 0: No data consistency error detected 1: Data consistency error detected This bit is cleared by setting CSIHnSTCR0.CSIHnDCEC. This bit can be written to when CSIHnCTL0.CSIHnPWR = 0. This bit is initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0. If this bit is set due to a data consistency error being detected and cleared by CSIHnSTCR0.CSIHnDCEC at the same time, setting the bit is prioritized.
1	CSIHnPE	Parity error flag 0: No parity error detected 1: Parity error detected This bit is cleared by setting CSIHnSTCR0.CSIHnPEC. This bit can be written to when CSIHnCTL0.CSIHnPWR = 0. This bit is initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0. If this bit is set due to a parity error being detected and cleared by CSIHnSTCR0.CSIHnPEC at the same time, setting the bit is prioritized.
0	CSIHnOVE	Overrun error flag 0: No overrun error detected 1: Overrun error detected This bit is cleared by setting CSIHnSTCR0.CSIHnOVEC. This bit can be written to when CSIHnCTL0.CSIHnPWR = 0. This bit is initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0. This bit is fixed to 0 in the dual buffer mode. If this bit is set due to an overrun error being detected and cleared by CSIHnSTCR0.CSIHnOVEC at the same time, setting the bit is prioritized.

Table 25-25 Memory mode operation

Bit name	Bit position	Direct access mode	FIFO mode	Transmit-only buffer mode	Dual buffer mode
CSIHnSRP[7:0]	31 to 24	Fixed to 0	Number of received data items	Fixed to 0	Fixed to 0
CSIHnSPF[7:0]	23 to 16	Fixed to 0	Number of data items that have not been transmitted	Fixed to 0	Fixed to 0
CSIHnTMOE	15	Fixed to 0	0: No error has been detected. 1: An error has been detected.	Fixed to 0	Fixed to 0
CSIHnOFE	14	Fixed to 0	0: No error has been detected. 1: An error has been detected.	Fixed to 0	Fixed to 0
CSIHnTSF	7	0: Idle state 1: Transmitting or preparing to transmit			
CSIHnFLF	5	Fixed to 0	0: Not full 1: Full	Fixed to 0	Fixed to 0
CSIHnEMF	4	Fixed to 1	0: Not empty 1: Empty	Fixed to 1	Fixed to 1
CSIHnDCE	3	0: No error has been detected. 1: An error has been detected.			
CSIHnPE	1	0: No error has been detected. 1: An error has been detected.			
CSIHnOVE	0	0: No error has been detected. 1: An error has been detected.			Fixed to 0

(5) CSIHnSTCR0 - CSIHn status clear register 0

This register clears the status flags of the CSIHnSTR0 status register.

Access This register can be written in 16-bit units.

When read, the value 0000_H is always returned.

Address <CSIHn_base> + 08_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHn TMOEC	CSIHn OFEC	0	0	0	0	0	CSIHn PCT	0	0	0	0	CSIHn DCEC	0	CSIHn PEC	CSIHn OVEC
W	W	R	R	R	R	R	W	R	R	R	R	W	R	W	W

Table 25-26 CSIHnSTCR0 register contents

Bit position	Bit name	Function
15	CSIHnTMOEC	Timeout error flag clear command 0: No operation. Read value is always 0. 1: Clear time out error flag (CSIHnSTR0.CSIHnTMOE)
14	CSIHnOFEC	Overflow error flag clear command 0: No operation. Read value is always 0. 1: Clear overflow error flag (CSIHnSTR0.CSIHnOFE)
8	CSIHnPCT	Controls the FIFO buffer pointers. 0: No operation. Read value is always 0. 1: In the dual buffer mode, transmit-only buffer mode, or FIFO mode, clear all the following FIFO buffer pointers: - CSIHnMRWP0.CSIHnTRWA[6:0] - CSIHnMRWP0.CSIHnRRA[6:0] - CSIHnMCTL2.CSIHnSOP[6:0] In only the FIFO mode, also clear all the following status bits: - CSIHnSTR0.CSIHnSPF[7:0] - CSIHnSTR0.CSIHnSRP[7:0] - CSIHnSTR0.CSIHnFLF - CSIHnSTR0.CSIHnTSF Note that CSIHnSTR0.CSIHnEMF is set (indicating an empty FIFO buffer). **Caution:** When this bit is set during communication, the communication stops.
3	CSIHnDCEC	Data consistency error flag clear command 0: No operation. Read value is always 0. 1: Clear data consistency error flag (CSIHnSTR0.CSIHnDCE)
1	CSIHnPEC	Parity error flag clear command 0: No operation. Read value is always 0. 1: Clear parity error flag (CSIHnSTR0.CSIHnPE)
0	CSIHnOVEC	Overrun error flag clear command 0: No operation. Read value is always 0. 1: Clear overrun error flag (CSIHnSTR0.CSIHnOVE)

(6) CSIHnMCTL0 - CSIHn memory control register 0

This register selects the memory mode and the timeout setting.

Access This register can be read or written in 16-bit units.

Address <CSIHn_base> + 1040_H

Initial Value $001F_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CSIHn MMS[1:0]		0	0	0	CSIHnTO[4:0]				
R	R	R	R	R	R	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W	R/W

Table 25-27 CSIHnMCTL0 register contents

Bit position	Bit name	Function
9, 8	CSIHn MMS[1:0]	Selects the memory mode. CSIHn MMS1 / CSIHn MMS0 / Description 0 / 0 / FIFO mode 0 / 1 / Dual buffer mode 1 / 0 / Transmit-only buffer mode 1 / 1 / Prohibited After changing the memory mode, set the CSIHnSTCR0.CSIHnPCT bit and clear the individual buffer pointers and other data. **Caution:** The memory mode can only be changed when CSIHnCTL0.CSIHnPWR and CSIHnCTL0.CSIHnMBS = 0.
4 to 0	CSIHn TO[4:0]	Select the FIFO mode timeout setting. CSIHnTO[4:0] / Description 00000_B / No timeout detection 00001_B / Timeout is (1 x 8 x BRG output clocks) 00010_B / Timeout is (2 x 8 x BRG output clocks) ... / 11111_B / Timeout is (31 x 8 x BRG output clocks) **Caution:** The timeout setting can only be changed when CSIHnCTL0.CSIHnPWR = 0. Clear these bits to 00000_B when in the master mode or a memory mode other than the FIFO mode (the direct access mode, dual buffer mode, or transmission mode). For details about timeout detection, see *(3) "Timeout error" on page 1877*.

(7) CSIHnMCTL1 - CSIHn memory control register 1

This register selects the conditions to generate the interrupt requests CSIHnTIC and CSIHnTIR in FIFO mode.

Access This register can be read or written in 32-bit units.

Address <CSIHn_base> + 1000_H

Initial Value $0000\ 007F_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	CSIHnFES[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	CSIHnFFS[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note This register can be written to during communication.

Table 25-28 CSIHnMCTL1 register contents

Bit position	Bit name	Function
22 to 16	CSIHnFES[6:0]	Select the condition for generating the CSIHnTIC interrupt (no transmission data). When the number of transmission data items in the FIFO buffer that have not been transmitted (checked by using the CSIHnSTR0.CSIHnSPF[7:0] bits) matches CSIHnMCTL1.CSIHnFES[6:0], a CSIHnTIC interrupt request is generated.
6 to 0	CSIHnFFS[6:0]	Select the condition for generating the CSIHnTIR interrupt (a full reception buffer). When the number of received data items in the FIFO buffer (checked by using the CSIHnSTR0.CSIHnSRP[7:0] bits) matches (128 – CSIHnMCTL1.CSIHnFFS[6:0]), a CSIHnTIR interrupt request is generated.

(8) CSIHnMCTL2 - CSIHn memory control register 2

This register controls memory operations while in the dual buffer mode or transmit-only buffer mode and generates triggers to start communication.

Access This register can be read or written in 32-bit units.

Address <CSIHn_base> + 1004_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. Writing to this register is prohibited when CSIHnSTR0.CSIHnTSF = 1 (during a transfer).
2. Writing to the CSIHnMCTL2 register is prohibited in the following cases:
   - When CSIHnCTL0.CSIHnPWR = 0
   - When CSIHnCTL0.CSIHnTXE = CSIHnCTL0.CSIHnRXE = 0
   - When in the direct access mode or FIFO mode

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSIHnBTST	0	0	0	0	0	0	0	CSIHnND[7:0]							
R/W	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	CSIHnSOP[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 25-29 CSIHnMCTL2 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>31</td><td>CSIHnBTST</td><td>Provides a start trigger for buffer transfer.
0: No operation
1: Start transfer command
The read value is always 0.
Caution:
This bit can only be used in dual buffer mode and transmit-only buffer mode.
In the direct access mode and FIFO mode, this bit is disabled.</td></tr>
<tr><td>23 to 16</td><td>CSIHnND[7:0]</td><td>Specify the number of data items.
When read, these bits indicate the number of remaining communication data items.
<table>
<tr><th>CSIHn ND[7:0]</th><th>Dual buffer mode</th><th>Transmit-only buffer mode</th><th>FIFO mode</th><th>Direct access mode</th></tr>
<tr><td>00_H</td><td>Transmit 0 data items.</td><td>Transmit 0 data items.</td><td>No effect</td><td>No effect</td></tr>
<tr><td>01_H</td><td>Transmit 1 data item.</td><td>Transmit 1 data item.</td><td>No effect</td><td>No effect</td></tr>
<tr><td>...</td><td>...</td><td>...</td><td>No effect</td><td>No effect</td></tr>
<tr><td>$3F_H$</td><td>Transmit 63 data items.</td><td>Transmit 63 data items.</td><td>No effect</td><td>No effect</td></tr>
<tr><td>40_H</td><td>Transmit 64 data items.</td><td>Transmit 64 data items.</td><td>No effect</td><td>No effect</td></tr>
<tr><td>...</td><td>Prohibited</td><td>...</td><td>No effect</td><td>No effect</td></tr>
<tr><td>$7F_H$</td><td>Prohibited</td><td>Transmit 127 data items.</td><td>No effect</td><td>No effect</td></tr>
<tr><td>80_H</td><td>Prohibited</td><td>Transmit 128 data items.</td><td>No effect</td><td>No effect</td></tr>
<tr><td>Other than the above</td><td colspan="4">Setting prohibited</td></tr>
</table>
The value of these bits is automatically decremented after transferring the data.
During a transfer, the number of remaining data items can be read from these bits.
The value of these bits is not decremented while in the direct access mode.</td></tr>
<tr><td>6 to 0</td><td>CSIHn SOP[6:0]</td><td>Select the transmission data pointer.
<table>
<tr><th>CSIHn SOP[6:0]</th><th>Dual buffer mode</th><th>Transmit-only buffer mode</th><th>FIFO mode</th><th>Direct access mode</th></tr>
<tr><td>00_H</td><td>0000_H</td><td>0000_H</td><td>0000_H</td><td>No effect</td></tr>
<tr><td>01_H</td><td>0004_H</td><td>0004_H</td><td>0004_H</td><td>No effect</td></tr>
<tr><td>...</td><td>...</td><td>...</td><td>...</td><td>No effect</td></tr>
<tr><td>$3F_H$</td><td>$00FC_H$</td><td>$00FC_H$</td><td>$00FC_H$</td><td>No effect</td></tr>
<tr><td>40_H</td><td>Prohibited</td><td>0100_H</td><td>0100_H</td><td>No effect</td></tr>
<tr><td>...</td><td>Prohibited</td><td>...</td><td>...</td><td>No effect</td></tr>
<tr><td>$7F_H$</td><td>Prohibited</td><td>$01FC_H$</td><td>$01FC_H$</td><td>No effect</td></tr>
</table>
When CSIHnCTL0.CSIHnPWR = 0 or CSIHnSTR0.CSIHnPCT is set to forcibly stop communication, these bits are cleared by the hardware.
Note:
In the FIFO mode, these bits indicate the transmission address. The value of these bits is not decremented while in the direct access mode.</td></tr>
</table>

(9) CSIHnMRWP0 - CSIHn memory read/write pointer register 0

This register sets the pointers for reading from and writing to the dual or transmit-only buffer.

Access This register can be read or written in 32-bit units.

Address <CSIHn_base> + 1018_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	CSIHnRRA[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	CSIHnTRWA[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution This register can be written to during communication.
Writing to this register in the direct access mode or FIFO mode is prohibited.

Table 25-30 CSIHnMRWP0 register contents (1/2)

Bit position	Bit name	Function
22 to 16	CSIHn RRA[6:0]	Selects the read pointer of the Rx buffer.

CSIHn RRA[6:0]	Dual buffer mode	Transmit-only buffer mode	FIFO mode	Direct access mode
00_H	0000_H	0000_H	0000_H	No effect
01_H	0004_H	0004_H	0004_H	No effect
...	...	...	...	No effect
$3F_H$	$00FC_H$	$00FC_H$	$00FC_H$	No effect
40_H	Prohibited	0100_H	0100_H	No effect
...	Prohibited	...	...	No effect
$7F_H$	Prohibited	$01FC_H$	$01FC_H$	No effect

These bits are automatically incremented when reception data is read.
If an overrun error occurs while reading the CSIHnRX0W or CSIHnRX0H register (when the CPU reads the CSIHnRX0W or CSIHnRX0H register while there is no data), the read pointer is not incremented.
These bits are cleared when CSIHnSTCR0.CSIHnPCT is set.
These bits are not incremented in the direct access mode or transmit-only buffer mode.
When writing in the transmit-only buffer mode, clear these bits to 0000_H.
In the FIFO mode, these bits indicate the read address of the reception data.

Table 25-30 CSIHnMRWP0 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>6 to 0</td><td>CSIHn TRWA[6:0]</td><td>Selects the read/write pointer of the Tx buffer.
<table>
<tr><th>CSIHn TRWA[6:0]</th><th>Dual buffer mode</th><th>Transmit-only buffer mode</th><th>FIFO mode</th><th>Direct access mode</th></tr>
<tr><td>00_H</td><td>0000_H</td><td>0000_H</td><td>0000_H</td><td>No effect</td></tr>
<tr><td>01_H</td><td>0004_H</td><td>0004_H</td><td>0004_H</td><td>No effect</td></tr>
<tr><td>...</td><td>...</td><td>...</td><td>...</td><td>No effect</td></tr>
<tr><td>$3F_H$</td><td>$00FC_H$</td><td>$00FC_H$</td><td>$00FC_H$</td><td>No effect</td></tr>
<tr><td>40_H</td><td>Prohibited</td><td>0100_H</td><td>0100_H</td><td>No effect</td></tr>
<tr><td>...</td><td>Prohibited</td><td>...</td><td>...</td><td>No effect</td></tr>
<tr><td>$7F_H$</td><td>Prohibited</td><td>$01FC_H$</td><td>$01FC_H$</td><td>No effect</td></tr>
</table>
When transmission data is read or written from the CPU, these bits are automatically incremented.
These bits are cleared when CSIHnSTCR0.CSIHnPCT is set.
In the direct access mode, these bits are not incremented.
In the FIFO mode, these bits indicate the read/write address of the transmission data.</td></tr>
</table>

(10) CSIHnCFGx - CSIHn configuration register x

These eight registers specify for each chip select signal CSIHnTCSSx prescaler, parity, data length, recessive configuration for broadcasting, serial data direction, clock phase and data phase, idle enforcement configuration, idle timing, hold timing, inter-data time, and setup timing.

Slave mode

In slave mode the transmission protocol setting of the CSIHnCFG0 register are relevant:

- CSIHnPS0: Parity usage
- CSIHnDLS0: Data length selection
- CSIHnDIR0: Data direction
- CSIHnCKP0, CSIHnDAP0: Clock phase and data phase

In the slave mode, clear the CSIHnCFG0 register bits other than the above and the CSIHnCFG1 to CSIHnCFG7 registers to 0.

Access

This register can be read or written in 32-bit units.

Address

CSIHnCFG0: <CSIHn_base> + 1044_H
CSIHnCFG1: <CSIHn_base> + 1048_H
CSIHnCFG2: <CSIHn_base> + $104C_H$
CSIHnCFG3: <CSIHn_base> + 1050_H
CSIHnCFG4: <CSIHn_base> + 1054_H
CSIHnCFG5: <CSIHn_base> + 1058_H
CSIHnCFG6: <CSIHn_base> + $105C_H$
CSIHnCFG7: <CSIHn_base> + 1060_H

Initial Value

$0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSIHn PSCLx[1:0]		CSIHn PSx[1:0]		CSIHnDLSx[3:0]				0	0	0	0	CSIHn RCBx	CSIHn DIRx	CSIHn CKPx	CSIHn DAPx
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHn IDLx	CSIHnIDx[2:0]			CSIHnHDx[3:0]				CSIHnINx[3:0]				CSIHnSPx[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution

Writing is only possible when CSIHnCTL0.CSIHnPWR = 0 (except when writing the same value, which is possible even when CSIHnCTL0.CSIHnPWR = 1).

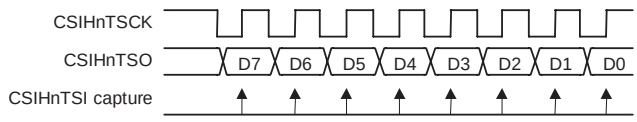
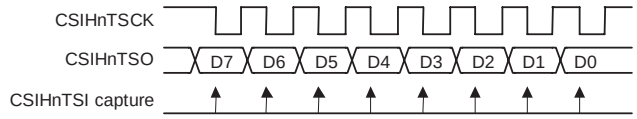
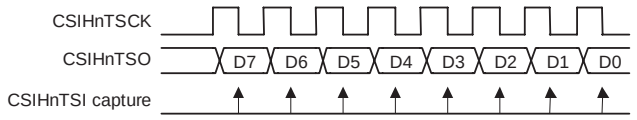
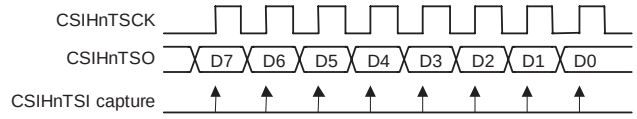
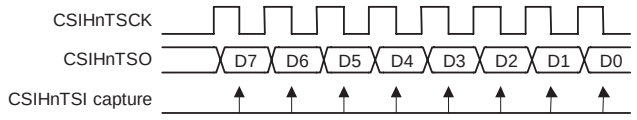
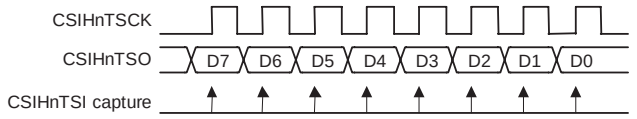
Table 25-31 CSIHnCFGx register contents (1/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>31 and 30</td><td>CSIHn PSCLx[1:0]</td><td>Selects the prescaler for chip select x.
<table>
<tr><th>CSIHn PSCLx1</th><th>CSIHn PSCLx0</th><th>Prescaler output</th></tr>
<tr><td>0</td><td>0</td><td>CSIHnBPCLK</td></tr>
<tr><td>0</td><td>1</td><td>CSIHnBPCLK / 2</td></tr>
<tr><td>1</td><td>0</td><td>CSIHnBPCLK / 4</td></tr>
<tr><td>1</td><td>1</td><td>CSIHnBPCLK / 8</td></tr>
</table>
These bits are only available in master mode.
For details about CSIHnBPCLK, see 25.3.6 “Serial clock selection” on page 1851.</td></tr>
<tr><td>29 and 28</td><td>CSIHn PSx[1:0]</td><td>Selects the parity for chip select x for transmission and reception.
<table>
<tr><th>CSIHn PSx1</th><th>CSIHn PSx0</th><th>Transmission</th><th>Reception</th></tr>
<tr><td>0</td><td>0</td><td>No parity transmitted</td><td>Parity reception is not expected.</td></tr>
<tr><td>0</td><td>1</td><td>Add parity bit fixed at 0</td><td>Parity bit reception is expected, but parity judgment is not performed.</td></tr>
<tr><td>1</td><td>0</td><td>Add odd parity</td><td>Odd parity bit reception is expected.</td></tr>
<tr><td>1</td><td>1</td><td>Add even parity</td><td>Even parity bit reception is expected.</td></tr>
</table></td></tr>
<tr><td>27 to 24</td><td>CSIHn DLSx[3:0]</td><td>Selects the data length for chip select x.
<table>
<tr><th>CSIHn DLSx[3:0]</th><th>Data length</th></tr>
<tr><td>0000_B</td><td>16 bits</td></tr>
<tr><td>0001_B</td><td>1 bit</td></tr>
<tr><td>0010_B</td><td>2 bits</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1111_B</td><td>15 bits</td></tr>
</table>
Note:
For details about the CSIHnDLSx[3:0] bit setting, see 25.3.9 “Data length selection”.
For the CSIHnDLSx[3:0] bits, 0001_B (1 bit) to 0110_B (6 bits) can be specified only when the data length is 16 bits or more.</td></tr>
<tr><td>19</td><td>CSIHn RCBx</td><td>Selects the recessive configuration for broadcasting for chip select x.
0: Dominant (higher priority)
1: Recessive (lower priority)
For details, see (1) “Configuration registers” on page 1846.</td></tr>
<tr><td>18</td><td>CSIHn DIRx</td><td>Selects the serial data direction for chip select x.
0: Data is sent/received with MSB first
1: Data is sent/received with LSB first
For details, see 25.3.10 “Serial data direction selection” on page 1859</td></tr>
</table>

Table 25-31 CSIHnCFGx register contents (2/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>17 and 16</td><td>CSIHn CKPx

CSIHn DAPx</td><td>CKP: Clock phase select bit
DAP: Data phase select bit

CSIHnCTL1.CSIHnCKR = 0
<table>
<tr><th>CSIHn CKPx</th><th>CSIHn DAPx</th><th>Clock phase and data phase selection</th></tr>
<tr><td>0</td><td>0</td><td> 
</td></tr>
<tr><td>0</td><td>1</td><td> 
</td></tr>
<tr><td>1</td><td>0</td><td> 
</td></tr>
<tr><td>1</td><td>1</td><td> 
</td></tr>
</table>
CSIHnCTL1.CSIHnCKR = 1
<table>
<tr><th>CSIHn CKPx</th><th>CSIHn DAPx</th><th>Clock phase and data phase selection</th></tr>
<tr><td>0</td><td>0</td><td> 
</td></tr>
<tr><td>0</td><td>1</td><td> 
</td></tr>
<tr><td>1</td><td>x</td><td>Setting prohibited</td></tr>
</table>
Caution:
When not using the chip select function, fix the CSIHnCKPx bit to 0, and use the CSIHnCTL1.CSIHnCKR bit to specify the clock phase.</td></tr>
<tr><td>15</td><td>CSIHn IDLx</td><td>Selects the idle enforcement configuration for chip select x.
0: If the chip select value did not change, the chip select signal stays active. If a different chip select value is defined, chip select signal x becomes idle.
1: An idle state is inserted after every transfer to chip select x.

This bit is only available in master mode.
If CSIHnCTL1.CSIHnJE = 1 and CSIHnTX0W.CSIHnEOJ = 1, chip select signal x definitely becomes idle even if CSIHnCFG0-7.CSIHnIDLn is cleared to 0.
For details about the idle state, see Figure 25-6 “Chip select timings” on page 1847.</td></tr>
</table>

Table 25-31 CSIHnCFGx register contents (3/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>14 to 12</td><td>CSIHn IDx[2:0]</td><td>Selects the idle time for chip select x.
<table>
<tr><th>CSIHnDLSx[3:0]</th><th>Idle timing</th></tr>
<tr><td>000_B</td><td>0.5 transmission clock cycles</td></tr>
<tr><td>001_B</td><td>1 transmission clock cycle</td></tr>
<tr><td>010_B</td><td>1.5 transmission clock cycles</td></tr>
<tr><td>...</td><td>... (2.5, 3.5, 4.5, 6.5)</td></tr>
<tr><td>111_B</td><td>8.5 transmission clock cycles</td></tr>
</table>
These bits are only available in master mode.</td></tr>
<tr><td>11 to 8</td><td>CSIHn HDx[3:0]</td><td>Selects the hold time for chip select x in transmission clock cycles.
<table>
<tr><th>CSIHnINx[3:0]</th><th>Hold timing with CSIHnCTL1.CSIHnSIT = 0</th><th>Hold timing with CSIHnCTL1.CSIHnSIT = 1</th></tr>
<tr><td>0000_B</td><td>0.5 serial clock cycles</td><td>1.0 serial clock cycles</td></tr>
<tr><td>0001_B</td><td>1 serial clock cycle</td><td>1.5 serial clock cycles</td></tr>
<tr><td>0010_B</td><td>1.5 serial clock cycles</td><td>2.0 serial clock cycles</td></tr>
<tr><td>...</td><td>... (2.5, 3.5, 4.5, 6.5, 8.5, 9.5 10.5, 11.5, 12.5, 14.5, 16.5, 18.5)</td><td>... (3.0, 4.0, 5.0, 7.0, 9.0, 10.0, 11.0, 12.0, 13.0, 15.0, 17.0, 19.0)</td></tr>
<tr><td>1111_B</td><td>20.5 serial clock cycles</td><td>21.0 serial clock cycles</td></tr>
</table>
These bits are only available in master mode.</td></tr>
</table>

Table 25-31 CSIHnCFGx register contents (4/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7 to 4</td><td>CSIHn INx[3:0]</td><td>Selects the inter-data time for chip select x in transmission clock cycles.
<table>
<tr><th>CSIHnINx[3:0]</th><th>Inter-data time when CSIHnCTL1.CSIHnSIT = 0</th><th>Inter-data time when CSIHnCTL1.CSIHnSIT = 1</th></tr>
<tr><td>0000_B</td><td>0.0 serial clock cycles</td><td>0.5 serial clock cycles</td></tr>
<tr><td>0001_B</td><td>0.5 serial clock cycles</td><td>1.0 serial clock cycles</td></tr>
<tr><td>0010_B</td><td>1.0 serial clock cycles</td><td>1.5 serial clock cycles</td></tr>
<tr><td>0011_B</td><td>2.0 serial clock cycles</td><td>2.5 serial clock cycles</td></tr>
<tr><td>...</td><td>... (3.0, 4.0, 6.0, 8.0, 9.0, 10.0, 11.0, 12.0, 14.0, 16.0, 18.0)</td><td>... (3.5, 4.5, 6.5, 8.5, 9.5, 10.5, 11.5 12.5, 14.5, 16.5, 18.5)</td></tr>
<tr><td>1111_B</td><td>20.0 serial clock cycles</td><td>20.5 serial clock cycles</td></tr>
</table>
These bits are only available in master mode.</td></tr>
<tr><td>3 to 0</td><td>CSIHn SPx[3:0]</td><td>Selects the setup time for chip select x in transmission clock cycles.
<table>
<tr><th>CSIHnSPx[3:0]</th><th>Setup delay</th></tr>
<tr><td>0000_B</td><td>0.5 serial clock cycles</td></tr>
<tr><td>0001_B</td><td>1.0 serial clock cycles</td></tr>
<tr><td>0010_B</td><td>1.5 serial clock cycles</td></tr>
<tr><td>...</td><td>... (2.5, 3.5, 4.5, 6.5, 8.5, 9.5 10.5, 11.5, 12.5, 14.5, 16.5, 18.5)</td></tr>
<tr><td>1111_B</td><td>20.5 serial clock cycles</td></tr>
</table>
These bits are only available in master mode.</td></tr>
</table>

(11) CSIHnTX0W - CSIHn transmit data register 0 for word access

This register stores the transmission data. In addition, it specifies the communication interrupt request, the end-of-job, the extended data length, and the chip select activation.

Access This register can be read or written in 32-bit units.

Address <CSIHn_base> + 1008_H

Initial Value Undefined

Cautions

1. Reading this register is prohibited during communication in the FIFO mode.
2. Reading from and writing to this register are prohibited in the FIFO mode when CSIHnCTL0.CSIHnPWR = 0.
3. Writing to this register is prohibited in the direct access mode when CSIHnCTL0.CSIHnTXE = CSIHnCTL0.CSIHnRXE = 0.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSIHnCIRE	CSIHnEOJ	CSIHnEDL	0	0	0	0	0	CSIHnCS[7:0]							
R/W	R/W	R/W	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnTX[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 25-32 CSIHnTX0W register contents (1/2)

Bit position	Bit name	Function
31	CSIHnCIRE	Enables the communication interrupt request CSIHnTIC in dual or transmit-only buffer mode or the job completion interrupt CSIHnTJC request in FIFO mode. 0: No interrupt requested 1: Interrupt requested. Generates interrupt CSIHnTIC or CSIHnTJC after transmission. For details, see *"CSIHnTIC in direct access mode" on page 1862* and *(4) "CSIHnTIJC job completion interrupt" on page 1869*. **Caution:** This bit is only valid when the job mode is enabled (CSIHnCTL1.CSIHnJE = 1).
30	CSIHnEOJ	Specifies the end of a job. 0: No end-of-job data 1: End-of-job data **Caution:** This bit is only valid when the job mode is enabled (CSIHnCTL1.CSIHnJE = 1). For use in the slave mode, clear this bit.

Table 25-32 CSIHnTX0W register contents (2/2)

Bit position	Bit name	Function
29	CSIHnEDL	Specifies whether the associated data requires the extended data length (EDL) option. 0: Normal operation 1: Extended data length activated The associated data is transmitted as of 16 bits. The inter-data delay time and idle time are not inserted after data transmission. When CSIHnCTL1.CSIHnEDLE = 1 and CSIHnTX0W.CSIHnEDL = 1, the same CS must also be selected for the second data. If the CS for the second data is changed, the correct operation is not guaranteed. **Caution:** This bit can only be used when CSIHnCTL1.CSIHnEDLE = 1.
23 to 16	CSIHnCSx	Activates one or several chip select signals. 0: Chip select x is activated for the associated transmission 1: Chip select x is deactivated for the associated transmission Setting CSIHnTX0W.CSIHnCS[7:0] to FF_H is prohibited. **Caution:** If several chip select signals are enabled for broadcasting, the configuration of one with CSIHnCFGx.CSIHnRCBx = 0 (dominant) is used. In this case, all dominant chip selects must be set to precisely the same configuration. For use in the slave mode, set the CSIHnCS[7:0] bits to FE_H.
15 to 0	CSIHnTX[15:0]	Stores the transmission data.

(12) CSIHnTX0H - CSIHn transmit data register 0 for half word access

This register stores the transmission data. This register is the same as bits 15 to 0 of register CSIHnTX0W.

Access This register can be read or written in 16-bit units.

Address <CSIHn_base> + $100C_H$

Initial Value Undefined

Cautions

1. Reading this register is prohibited during communication in the FIFO mode.
2. Reading from and writing to this register are prohibited in the FIFO mode when CSIHnCTL0.CSIHnPWR = 0.
3. Writing to this register is prohibited in the direct access mode when CSIHnCTL0.CSIHnTXE = CSIHnCTL0.CSIHnRXE = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnTX[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 25-33 CSIHnTX0H register contents

Bit position	Bit name	Function
15 to 0	CSIHnTX[15:0]	Stores the transmission data.

(13) CSIHnRX0W - CSIHn receive data register 0 for word access

This register stores the received data.

Access This register is read-only, in 32-bit units.

Address <CSIHn_base> + 1010_H

Initial Value Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	CSIHnRPE	CSIHnTDCE	CSIHnCS[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnRX[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Cautions

1. This register can be read when CSIHnCTL0.CSIHnPWR = 1, and can be written to when CSIHnCTL0.CSIHnPWR = 0.
2. This register is Initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0.
3. Reading from and writing to this register are prohibited in the FIFO mode when CSIHnCTL0.CSIHnPWR = 0.

Table 25-34 CSIHnRX0W register contents

Bit position	Bit name	Function
25	CSIHnRPE	Indicates whether a reception data parity error was detected. 0: No parity error has been detected in the received data. 1: A parity error has been detected in the received data.
24	CSIHnTDCE	Indicates whether a transmission data consistency error was detected. A data consistency check is performed on transmission data. The result of the check performed on the data transmitted at the same time as saving received data to CSIHnRX0W.CSIHnRX[15:0] is applied to this bit. 0: No data consistency error has been detected in the transmitted data. 1: A data consistency error has been detected in the transmitted data.
23 to 16	CSIHnCSx	Indicate whether the chip select signal is active. When in the master mode, the status of the chip select signal upon receiving the data saved to CSIHnRX0W.CSIHnRX[15:0] (that is, which CS to perform communication for) is stored in these bits. 0: Chip select signal x was active upon receiving the data. 1: Chip select signal x was inactive upon receiving the data. When in the slave mode, because it is necessary to specify CS0 (CSIHnTX0W.CSIHnCS[7:0] = FE_H) as the communication partner when transmission is enabled, FE_H is saved in the transmission mode or transmission/reception mode. The value is always 00_H when in the reception mode.
15 to 0	CSIHnRX[15:0]	Store the reception data. Read the value of the CSIHnRX0W or CSIHnRX0H register at least one serial clock cycle before the interrupt is generated.

(14) CSIHnRX0H - CSIHn receive data register 0 for half word access

This register stores the reception data. This register is the same as bits 15 to 0 of register CSIHnTX0W.

Access This register is read-only, in 16-bit units.

Address <CSIHn_base> + 1014_H

Initial Value Undefined

Cautions

1. This register can be read when CSIHnCTL0.CSIHnPWR = 1, and can be written to when CSIHnCTL0.CSIHnPWR = 0.
2. This register is initialized when CSIHnCTL0.CSIHnPWR is changed from 0 to 1 or 1 to 0.
3. Reading from and writing to this register are prohibited in the FIFO mode when CSIHnCTL0.CSIHnPWR = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnRX[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 25-35 CSIHnRX0H register contents

Bit position	Bit name	Function
15 to 0	CSIHnRX[15:0]	Stores the reception data.

25.5 Operating Procedures

The following examples and instructions are sorted according to the memory mode:

- Direct access
- Transmit-only buffer
- Dual buffer
- FIFO

25.5.1 Procedures in direct access mode

(1) For transmission/reception in the master mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- A CSIHnTIC interrupt is generated when transferring starts. (CSIHnCTL1.CSIHnCLIT = 1)
- Direct access mode (CSIHnCTL0.CSIHnMBS = 0)

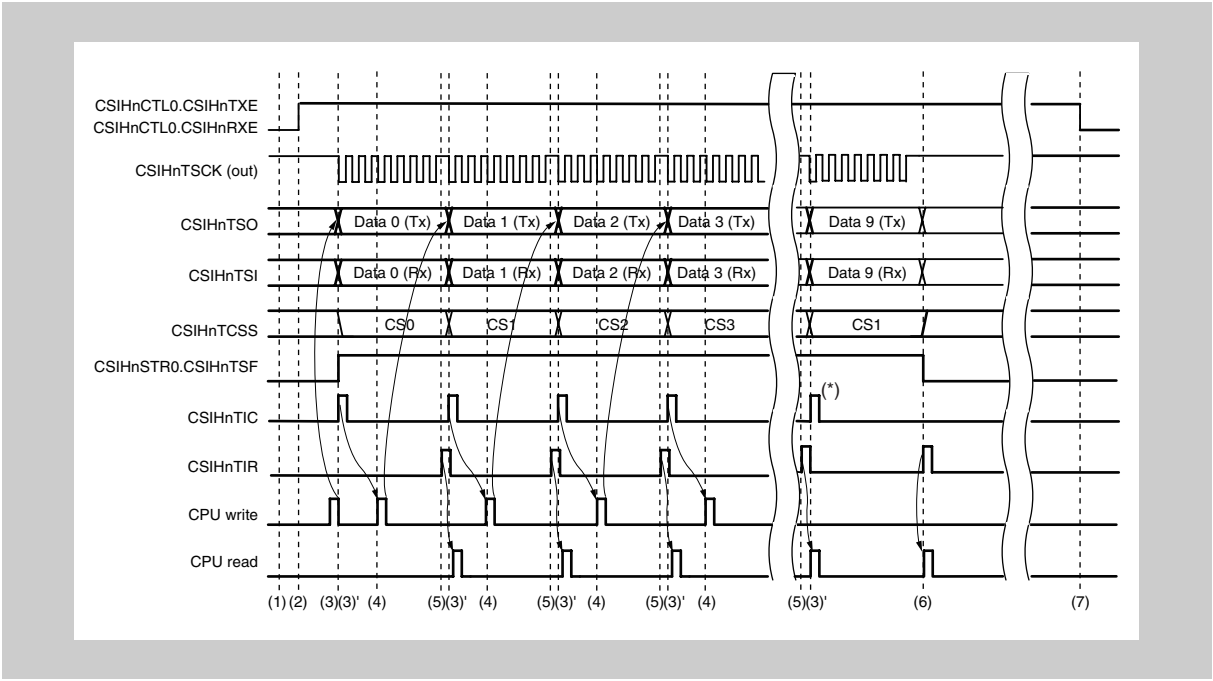


Figure 25-44 Direct access mode (for transmission/reception in the master mode, and when the job mode is disabled)

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS3 are used.)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 1 (direct access mode selected)
3. Write the first data to the transmission data register CSIHnTX0W. This write operation activates CS0, and transmission automatically starts.

3'. When CSIHnCTL1.CSIHnSLIT is set to 1, CSIHnTIC is generated at the start edge of CSIHnTSCK. CSIHnTIC indicates that the second data can be written to CSIHnTX0W.

4. Write the second data to CSIHnTX0W. If necessary, it is possible to change the CS and make a different device the communication partner. By writing the second data immediately after writing the first data, the unnecessary inter-data delay can be avoided.
5. Each time data is received, a CSIHnTIR interrupt is generated.
   – CSIHnTIR indicates that the reception data register CSIHnRX0 must be read.
6. If the CSIHnTIC interrupt indicated by “*” in the figure is the last one, it is not necessary to write to the transmission data register CSIHnTX0W based on the corresponding CSIHnTIC interrupt.
7. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption of the CSIH.

(2) For reception in the master mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- A CSIHnTIC interrupt is generated when transferring starts. (CSHICTL1.CSIHnSLIT = 1)
- Direct access mode (CSHICTL0.CSIHnMBS = 1)

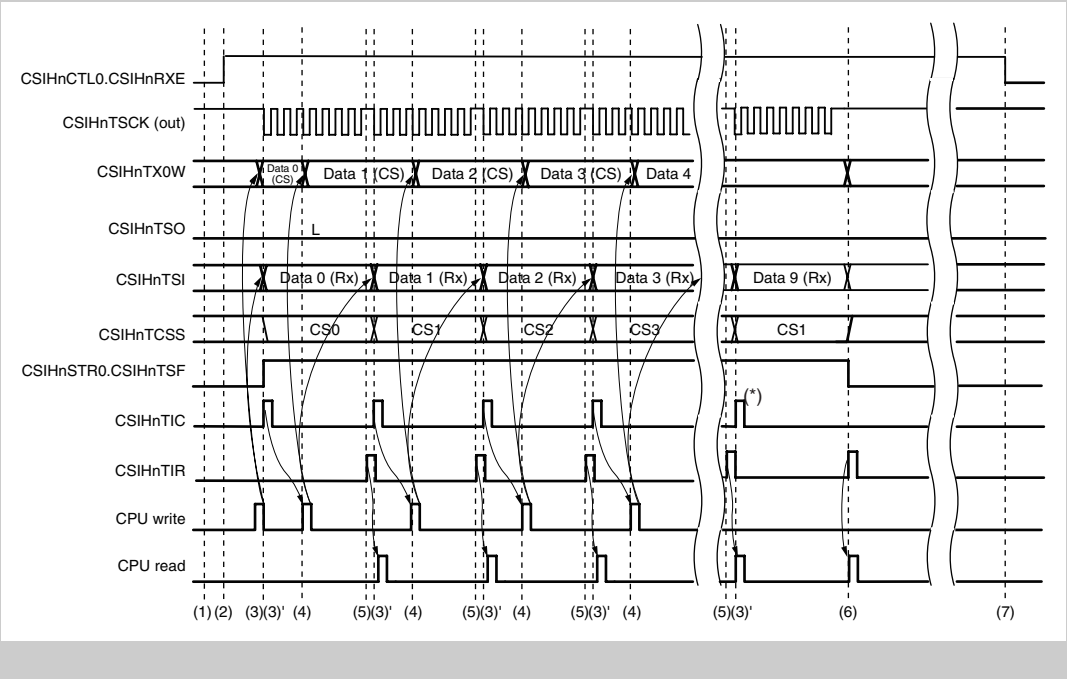


Figure 25-45 Direct access mode (for reception in the master mode, and when the job mode is disabled)

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS3 are used.)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 1 (direct access mode selected)
3. Write the transmission data[1] to the transmission data register CSIHnTX0W for the CS data. This write operation activates CS0, and reception automatically starts.

1) The transmission data is not used, but the chip select signal is enabled.

3'. When CSIHnCTL1.CSIHnSLIT is set to 1, CSIHnTIC is generated at the start edge of CSIHnTSCK. CSIHnTIC indicates that the second data can be written to CSIHnTX0W.

4. Write the second data to CSIHnTX0W. If necessary, it is possible to change the CS and make a different device the communication partner. By writing the second data immediately after writing the first data, the unnecessary inter-data delay can be avoided.

5. Each time data is received, a CSIHnTIR interrupt is generated.

   – CSIHnTIR indicates that the reception data register CSIHnRX0W must be read.

6. If the CSIHnTIC interrupt indicated by “*” in the figure is the last one, it is not necessary to write to the transmission data register CSIHnTX0W based on the corresponding CSIHnTIC interrupt.

7. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption of the CSIH.

(3) For transmission/reception in the slave mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFG0.CSIHnDLS0[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFG0.CSIHnDIR0 = 0)
- Normal clock phase and data phase (CSIHnCFG0.CSIHnCKP0 = 0, CSIHnCFG0.CSIHnDAP0 = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Handshake enabled (CSIHnCTL1.CSIHnHSE = 1)
- CSIHnTIC interrupt generated at the transfer start timing (CSIHnCTL1.CSIHnSLIT = 1)
- Direct access mode (CSIHnCTL0.CSIHnMBS = 1)

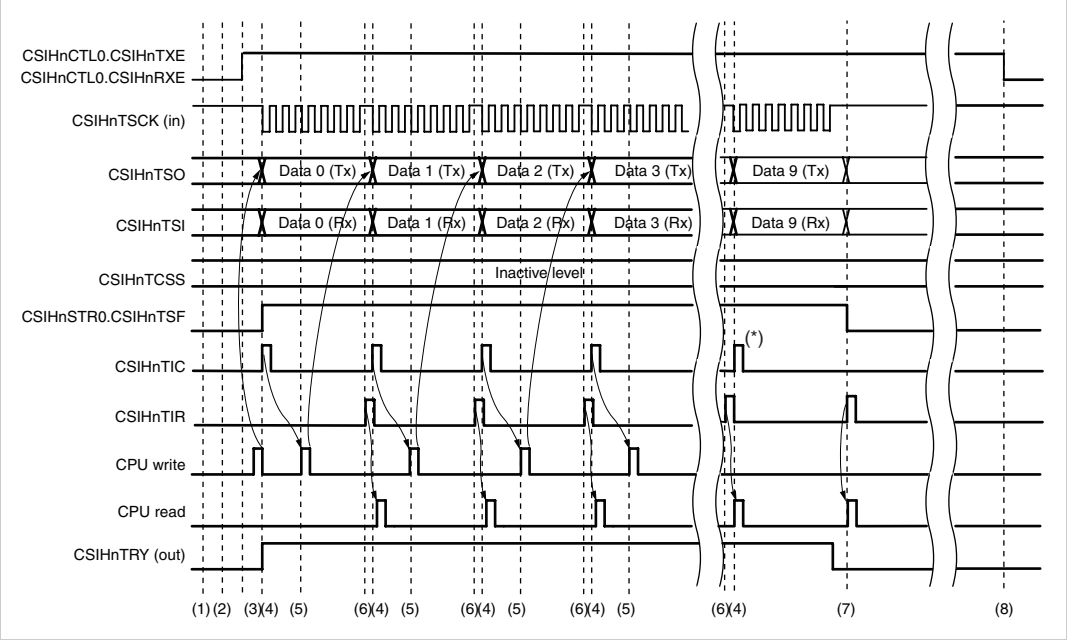


Figure 25-46 Direct access mode (for transmission/reception in the slave mode, and when the job mode is disabled)

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnCFG0 (communication protocol)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 1 (direct access mode selected)
3. Write the first data to the transmission data register CSIHnTX0W. The CSIHnTRY signal is switched from BUSY (low level) to READY (high level) by writing data. When a serial clock is supplied from the master, communication automatically starts.
4. When CSIHnCTL1.CSIHnSLIT is set to 1, CSIHnTIC is generated at the start edge of CSIHnTSCK. CSIHnTIC indicates that the second data can be written to CSIHnTX0W.

5. Write the second data to CSIHnTX0W. By writing the second data immediately after writing the first data, the unnecessary inter-data delay can be avoided.
6. Each time data is received, a CSIHnTIR interrupt is generated.
   – CSIHnTIR indicates that the reception data register CSIHnRX0W must be read.
7. If the CSIHnTIC interrupt indicated by "*" in the figure is the last one, it is not necessary to write to the transmission data register CSIHnTX0W based on the corresponding CSIHnTIC interrupt.
8. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption of the CSIH.

(4) For reception in the slave mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFG0.CSIHnDLS0[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFG0.CSIHnDIR0 = 0)
- Normal clock phase and data phase (CSIHnCFG0.CSIHnCKP0 = 0, CSIHnCFG0.CSIHnDAP0 = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Handshake enabled (CSIHnCTL1.CSIHnHSE = 1)
- CSIHnTIC interrupt generated at the transfer start timing (CSIHnCTL1.CSIHnSLIT = 1)
- Direct access mode (CSIHnCTL0.CSIHnMBS = 1)

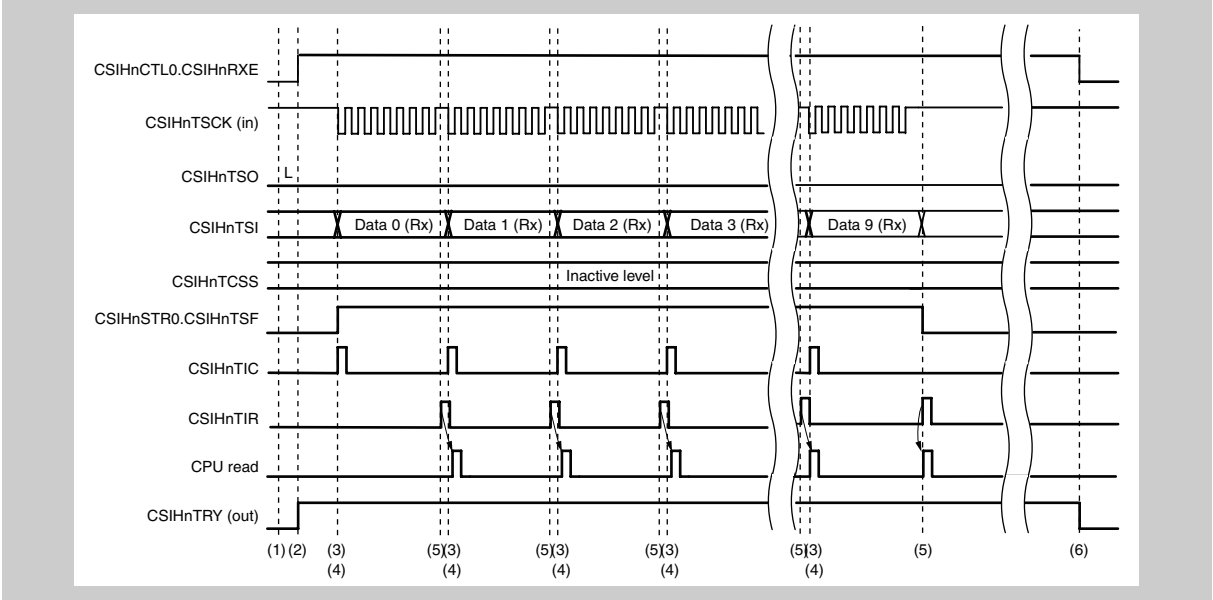


Figure 25-47 Direct access mode (for reception in the slave mode, and when the job mode is disabled)

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnCFG0 (communication protocol)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 1 (direct access mode selected)
3. When a serial clock is supplied from the master, reception automatically starts.
4. When CSIHnCTL1.CSIHnSLIT is set to 1, CSIHnTIC is generated at the start edge of CSIHnTSCK.
5. Each time data is received, a CSIHnTIR interrupt is generated.
   – CSIHnTIR indicates that the reception data register CSIHnRX0W must be read.
6. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption of the CSIH.

(5) For transmission/reception in the master mode, and when the job mode is enabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- CSIHnTIC interrupt generated at the transfer start timing (CSIHnCTL1.CSIHnSLIT = 1)
- Direct access mode (CSIHnCTL0.CSIHnMBS = 1)
- Two jobs that each transmit three data packets

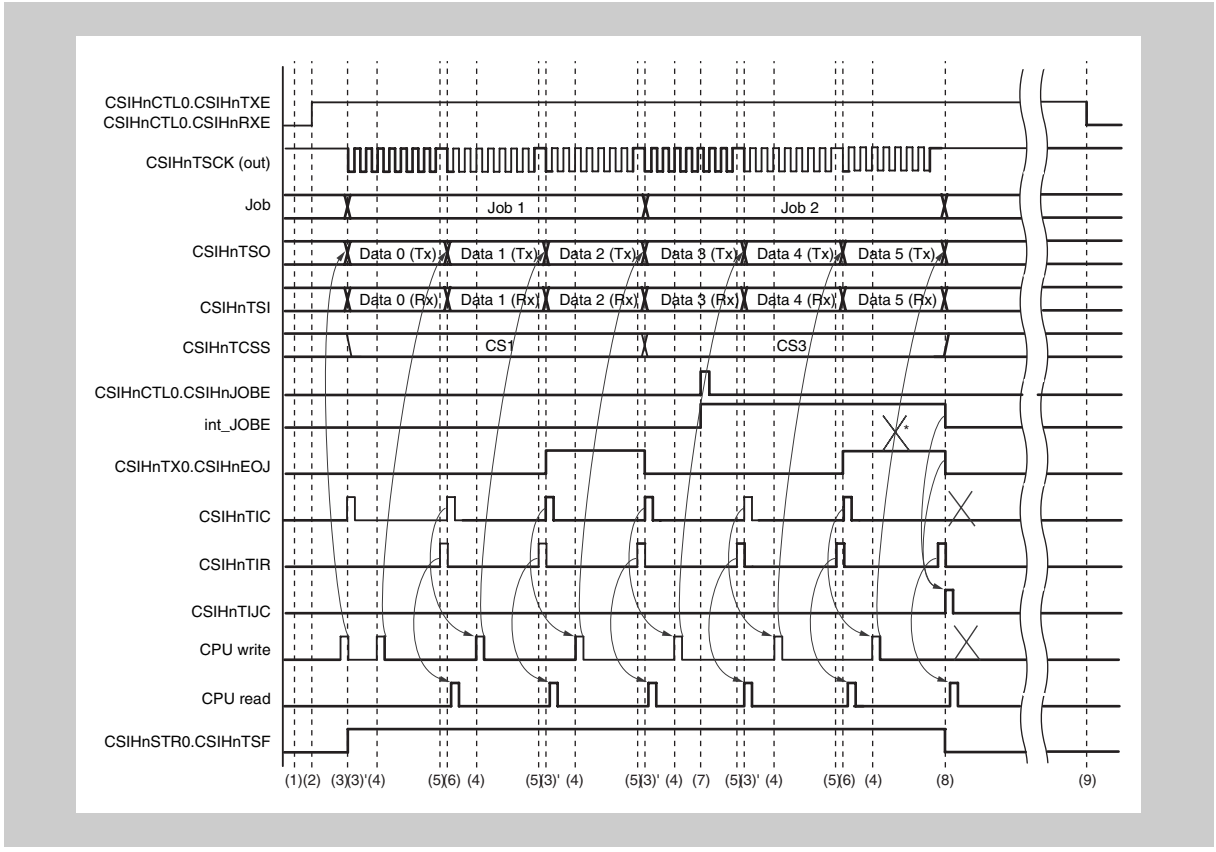


Figure 25-48 Direct access mode (for transmission/reception in the master mode, and when the job mode is enabled)

Note The int_JOBE signal in the above timing chart is the internal signal of the CSIHnJOBE bit.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS1 and CS3 are used.)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 1 (direct access mode selected)
3. Write the first transmission data packet to the transmission data register CSIHnTX0W. Transmission automatically starts when the first data can be used.

   The CSIHnSTR0.CSIHnTSF flag indicates that communication is being performed.

3'. When CSIHnCTL1.CSIHnSLIT is set to 1, CSIHnTIC is generated at the start edge of CSIHnTSCK. CSIHnTIC indicates that the second data can be written to CSIHnTX0W.

4. Write the second data to CSIHnTX0W. By writing the second data immediately after writing the first data, the unnecessary inter-data delay can be avoided.
5. Each time data is received, a CSIHnTIR interrupt request is generated.
   – CSIHnTIR indicates that the reception data register CSIHnRX0W must be read.
6. If the CSIHnTX0W register transfer data is the last job data, CSIHnTX0W.CSIHnEOJ is set to 1.
7. By setting CSIHnCTL0.CSIHnJOBE to 1, communication is forcibly stopped when the current job (job 2) ends.
8. After communication is forcibly stopped, the interrupt request CSIHnTIC is replaced with CSIHnTIJC. CSIHnTIR is generated as usual.

   The interrupt request CSIHnTIJC indicates that communication was forcibly stopped when the current job ended.
   The interrupt request CSIHnTIC is not generated. Note that the usable transmission data in the CSIHnTX0 register (indicated by “*” in the figure) is not transmitted.
9. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption of the CSIH.

(6) For reception in the master mode, and when the job mode is enabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- CSIHnTIC interrupt generated at the transfer start timing (CSIHnCTL1.CSIHnSLIT = 1)
- Direct access mode (CSIHnCTL0.CSIHnMBS = 1)
- Two jobs that each transmit three data packets

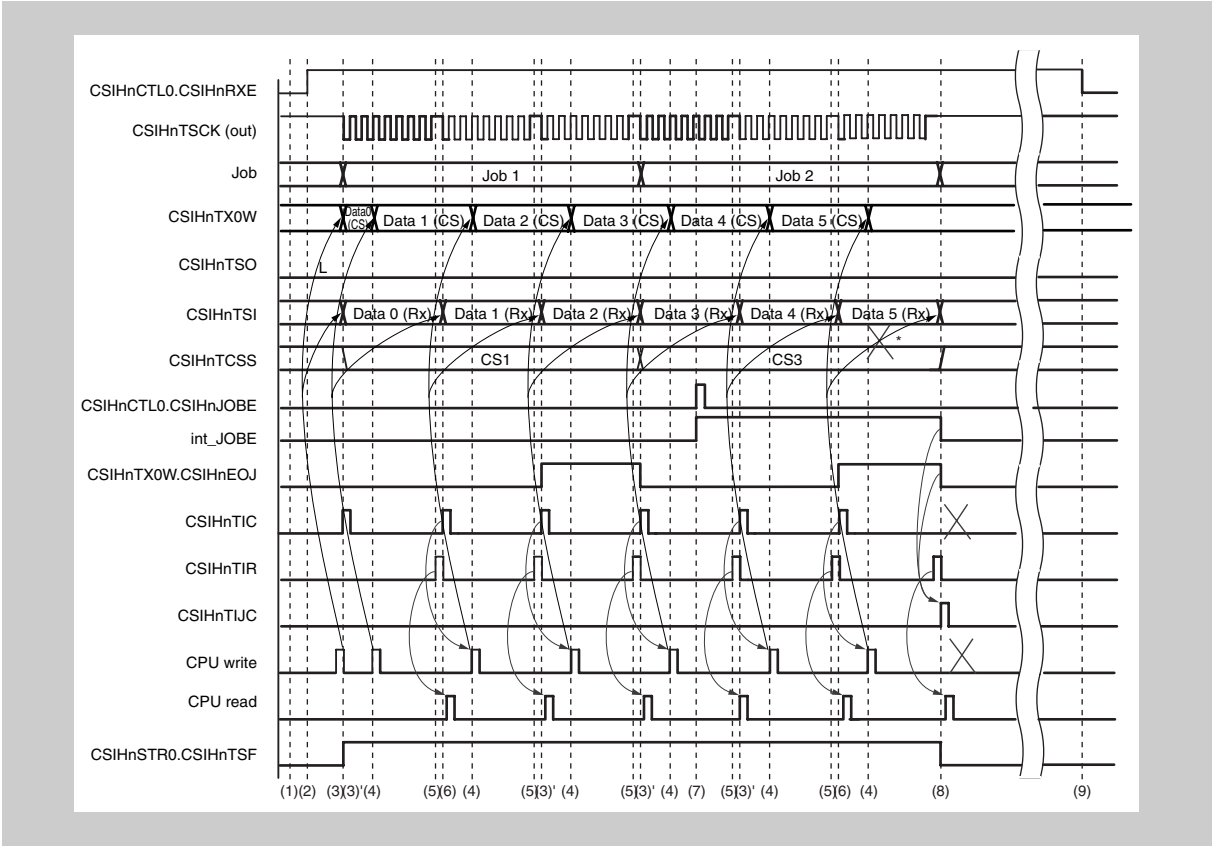


Figure 25-49 Direct access mode (for reception in the master mode, and when the job mode is enabled)

Note The int_JOBE signal in the above timing chart is the internal signal of the CSIHnJOBE bit.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS1 and CS3 are used.)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnMBS = 1 (direct access mode selected)
3. Write the transmission data to the transmission data register CSIHnTX0W for reception.
   Reception automatically starts. In addition, the CSIHnSTR0.CSIHnTSF bit is set.

3'. When CSIHnCTL1.CSIHnSLIT is set to 1, CSIHnTIC is generated at the start edge of CSIHnTSCK. CSIHnTIC indicates that the second data can be written to CSIHnTX0W.

4. Write the second data to CSIHnTX0H. By writing the second data immediately after writing the first data, the unnecessary inter-data delay can be avoided.
5. Each time data is received, a CSIHnTIR interrupt request is generated.
   – CSIHnTIR indicates that the reception data register CSIHnRX0 must be read.
6. If the CSIHnTX0W register transfer data is the last job data, CSIHnTX0W.CSIHnEOJ is set to 1.
7. By setting CSIHnCTL0.CSIHnJOBE to 1, communication is forcibly stopped when the current job (job 2) ends.
8. When int_JOBE is set and the last job 2 data is received, the interrupt request CSIHnTIJC is generated instead of CSIHnTIC.

   CSIHnTIR is generated as usual.
   The interrupt request CSIHnTIJC indicates that reception was forcibly stopped when the current job ended.
   The interrupt request CSIHnTIC is not generated. Note that the data indicated by “*” in the figure is not transferred.
9. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption of the CSIH.

25.5.2 Procedures in transmit-only buffer mode

This section provides examples where the job mode is enabled or disabled.

(1) For transmission/reception in the master mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Transmit-only buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 10_B)
- Number of transmitted data items: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

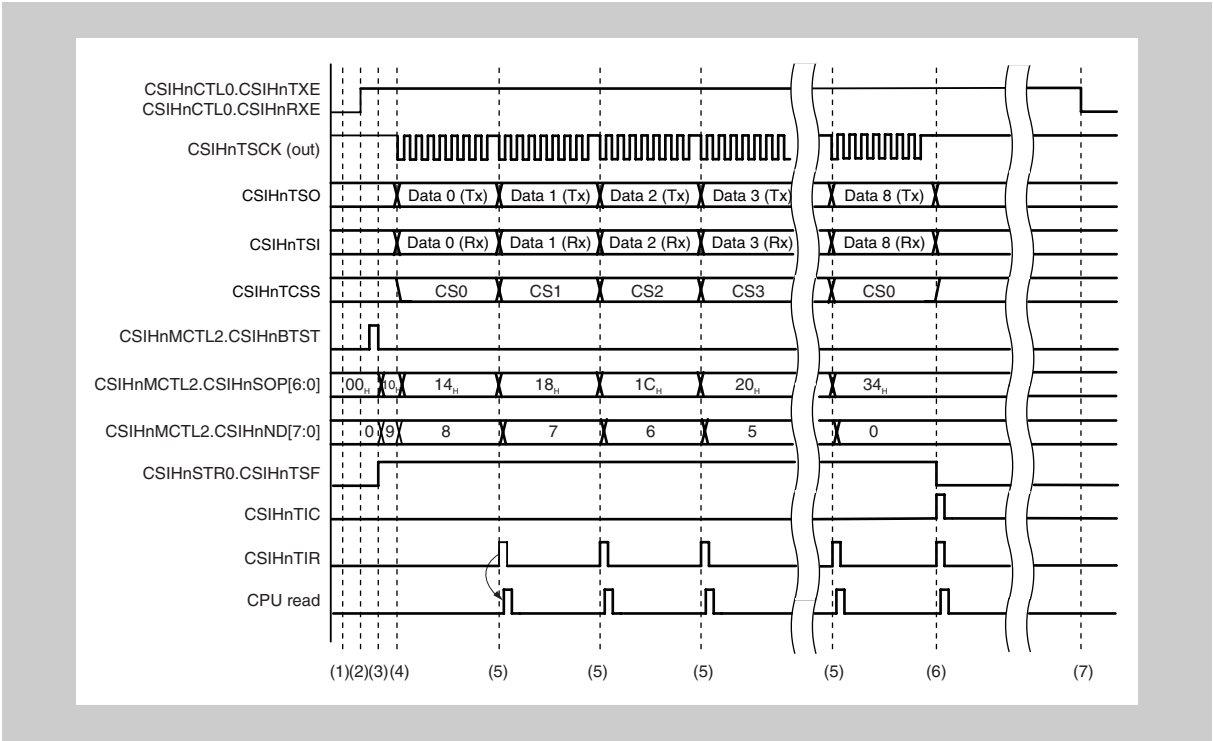


Figure 25-50 Transmit-only buffer mode (for transmission/reception in the master mode, and when the job mode is disabled)

Note The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 10_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS3 are used.)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits. Communication is started by setting CSIHnMCTL2.CSIHnBTST.
4. Transmission/reception starts. The CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented each time a data item is transmitted.
5. When all the data is received, CSIHnTIR is generated. The CSIHnTIR interrupt indicates that the reception data register CSIHnRX0W must be read.
6. When all the data is transmitted, a CSIHnTIC interrupt request is generated.
7. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(2) For reception in the master mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- Transmit-only buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 10_B)
- Number of transmitted data items: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

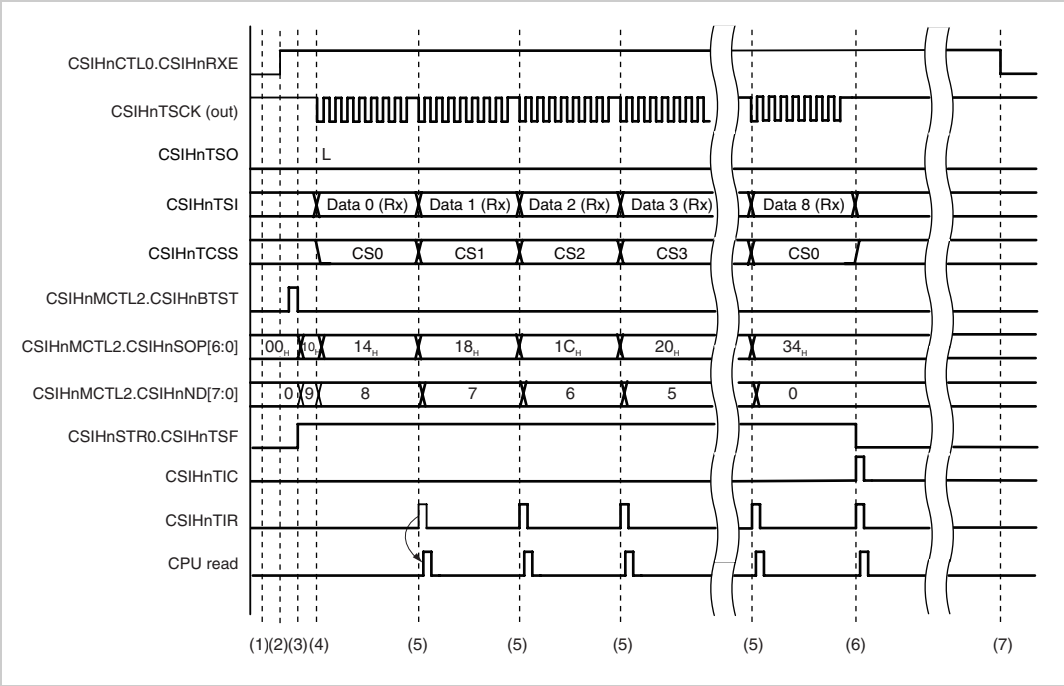


Figure 25-51 Transmit-only buffer mode (for reception in the master mode, and when the job mode is disabled)

Note The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 10_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS3 are used.)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits. Communication is started by setting CSIHnMCTL2.CSIHnBTST.
4. Reception starts. The CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented each time a data packet is transmitted.
5. When all the data is received, CSIHnTIR is generated. The CSIHnTIR interrupt indicates that the reception data register CSIHnRX0W must be read.
6. When all the data is received, a CSIHnTIC interrupt request is generated.
7. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(3) For transmission/reception in the slave mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFG0.CSIHnDLS0[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFG0.CSIHnDIR0 = 0)
- Normal clock phase and data phase (CSIHnCFG0.CSIHnCKP0 = 0, CSIHnCFG0.CSIHnDAP0 = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Handshake enabled (CSIHnCTL1.CSIHnHSE = 1)
- Transmit-only buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 10_B)
- Number of transmitted data items: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

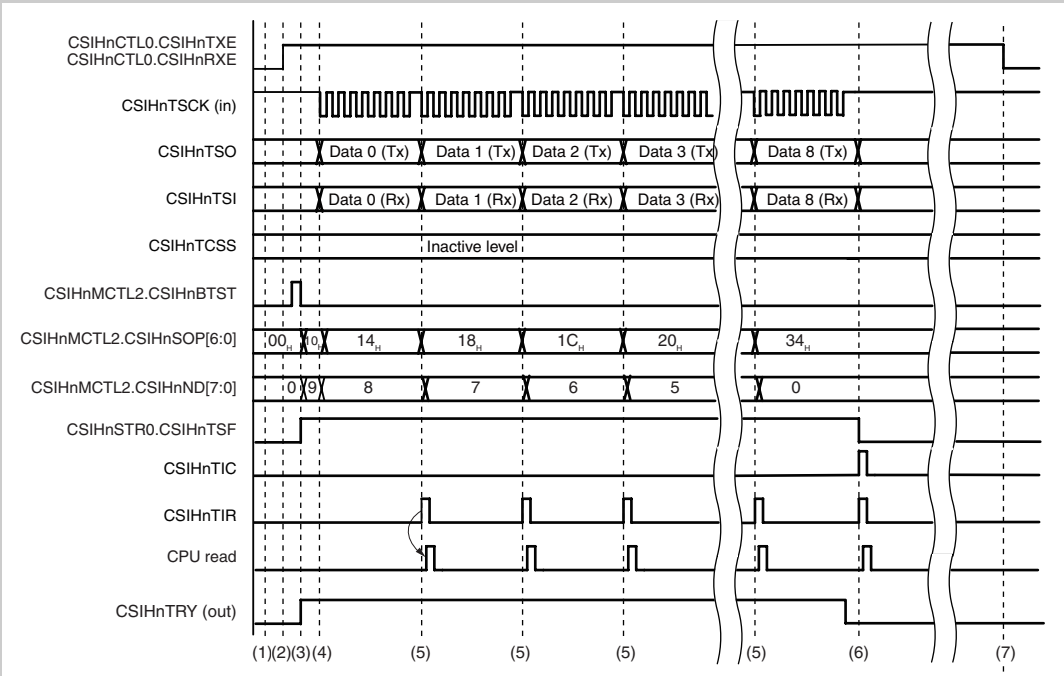


Figure 25-52 Transmit-only buffer mode (for transmission/reception in the slave mode, and when the job mode is disabled)

Note The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 10_B (memory mode)
   CSIHnCFG0 (communication protocol)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits. Communication is started by setting CSIHnMCTL2.CSIHnBTST.
4. When a serial clock is supplied from the master, communication starts. The CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented each time a data packet is transmitted.
5. Each time data is received, CSIHnTIR is generated. The CSIHnTIR interrupt indicates that the reception data register CSIHnRX0W must be read.
6. When all the data is received, a CSIHnTIC interrupt request is generated.
7. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(4) For reception in the slave mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFG0.CSIHnDLS0[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFG0.CSIHnDIR0 = 0)
- Normal clock phase and data phase (CSIHnCFG0.CSIHnCKP0 = 0, CSIHnCFG0.CSIHnDAP0 = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Handshake enabled (CSIHnCTL1.CSIHnHSE = 1)
- Transmit-only buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 10_B)
- Number of transmitted data items: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

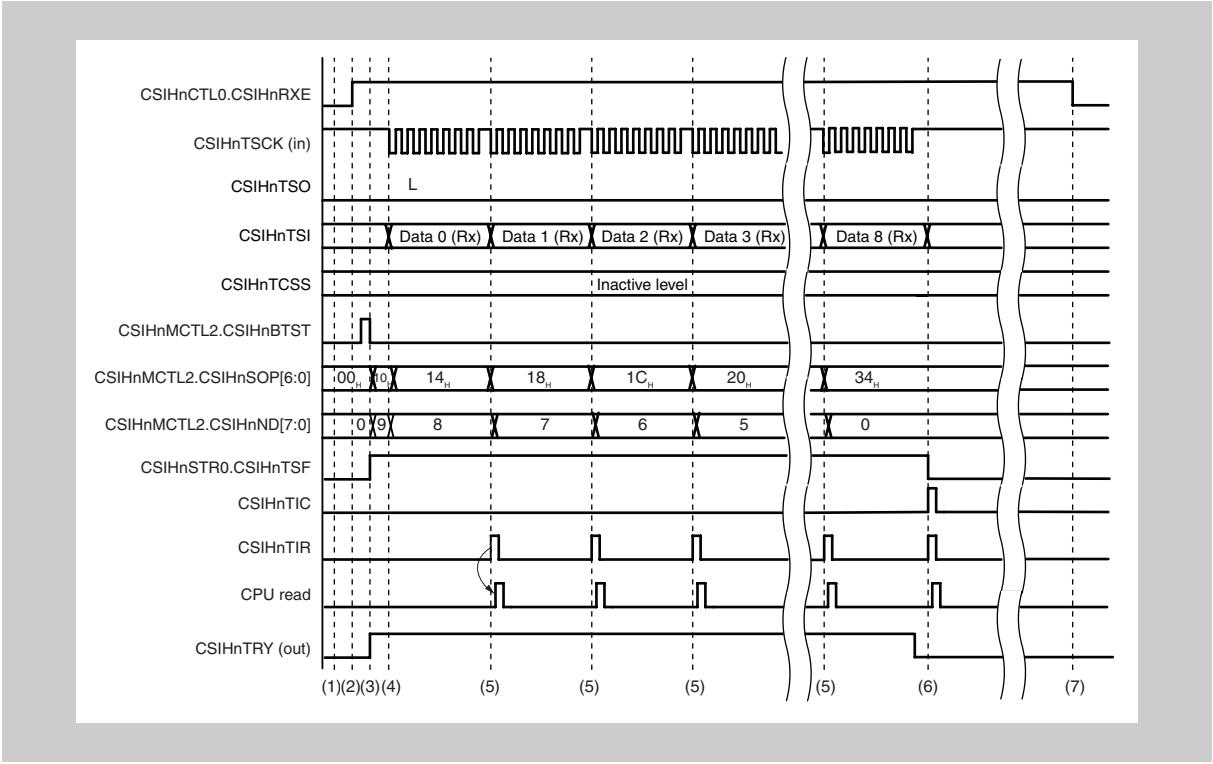


Figure 25-53 Transmit-only buffer mode (for reception in the slave mode, and when the job mode is disabled)

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 10_B (memory mode)
   CSIHnCFG0 (communication protocol)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits. Reception is started by setting CSIHnMCTL2.CSIHnBTST.
4. When a serial clock is supplied from the master, reception starts. The CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented each time a data packet is transmitted.
5. Each time data is received, CSIHnTIR is generated. The CSIHnTIR interrupt indicates that the reception data register CSIHnRX0W must be read.
6. When all the data is received, a CSIHnTIC interrupt request is generated.
7. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(5) For transmission/reception in the master mode, and when the job mode is enabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- Transmit-only buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 10_B)
- Number of transmitted data items: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

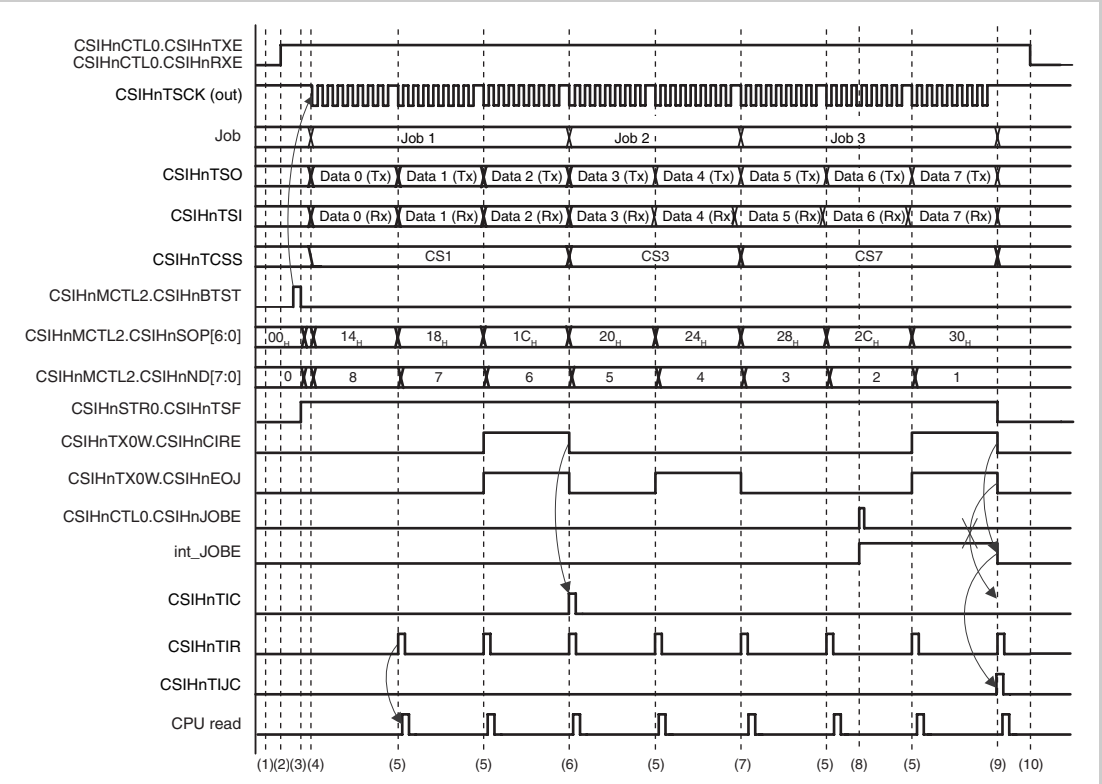


Figure 25-54 Transmit-only buffer mode (for transmission/reception in the master mode, and when the job mode is enabled)

Notes

1. The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.
2. The int_JOBE signal in the above timing chart is the internal signal of the CSIHnJOBE bit.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 10_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS1, CS3, and CS7 are used.)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits.
   Communication is started by setting CSIHnMCTL2.CSIHnBTST.
4. Transmission starts. The CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented each time a data item is transmitted.
5. Each time a data item is received, a CSIHnTIR interrupt request is generated.
   CSIHnTIR indicates that the reception data register CSIHnRX0W must be read.
6. CSIHnTIC is generated by setting CSIHnTX0W.CSIHnCIRE to 1.
   CSIHnTIC indicates that the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted.
7. Because the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted by clearing CSIHnTX0W.CSIHnCIRE, the interrupt request CSIHnTIC is not generated.
8. By setting CSIHnCTL0.CSIHnJOBE, communication is forcibly stopped when job 3 ends.
9. After communication is forcibly stopped, the interrupt requests CSIHnTIJC and CSIHnTIR are generated when job 3 ends.

   The interrupt request CSIHnTIJC indicates that communication was forcibly stopped when the current job ended.
   Because the interrupt request CSIHnTIJC is generated instead of the interrupt request CSIHnTIC, the interrupt request CSIHnTIC is not generated.
10. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(6) For reception in the master mode, and when the job mode is enabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- Transmit-only buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 10_B)
- Number of transmitted data items: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

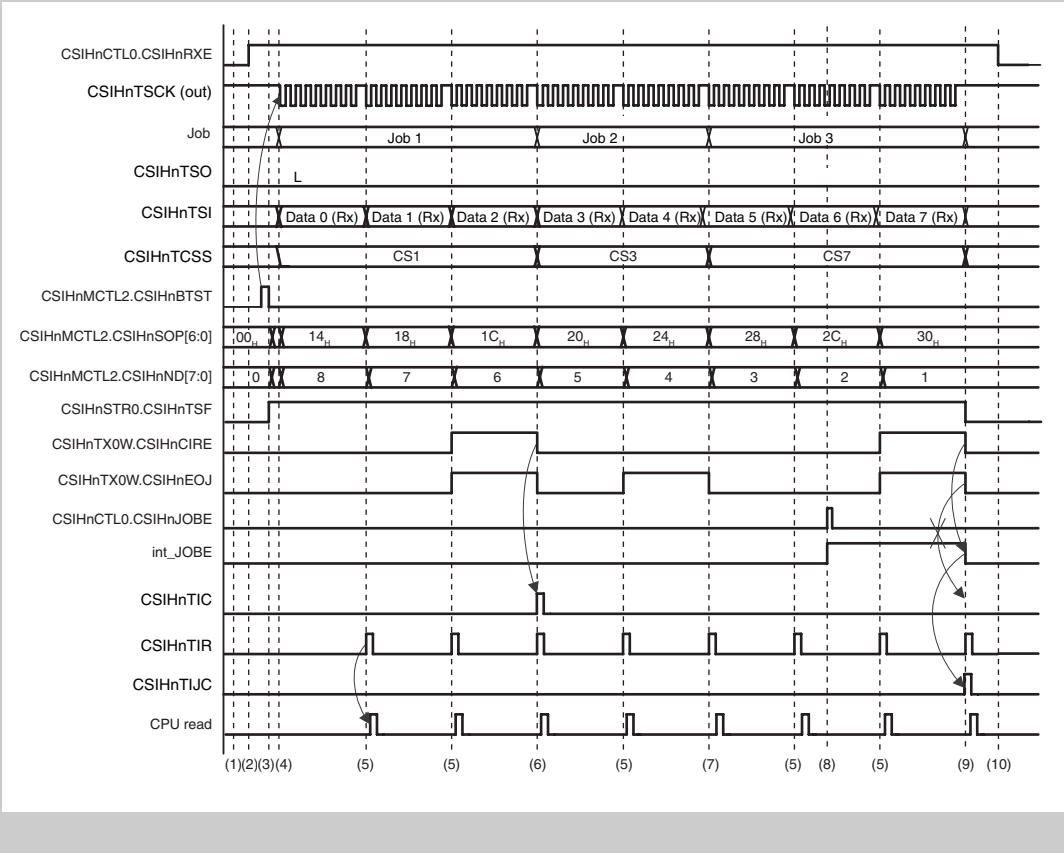


Figure 25-55 Transmit-only buffer mode (for reception in the master mode, and when the job mode is enabled)

Notes

1. The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.
2. The int_JOBE signal in the above timing chart is the internal signal of the CSIHnJOBE bit.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 10_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS1, CS3, and CS7 are used.)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits. Communication is started by setting CSIHnMCTL2.CSIHnBTST.
4. Reception starts. The CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented each time a data item is transmitted.
5. Each time a data item is received, a CSIHnTIR interrupt request is generated.
   CSIHnTIR indicates that the reception data register CSIHnRX0W must be read.
6. CSIHnTIC is generated by setting CSIHnTX0W.CSIHnCIRE to 1.
   CSIHnTIC indicates that the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted.
7. Because the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted by clearing CSIHnTX0W.CSIHnCIRE, the interrupt request CSIHnTIC is not generated.
8. By setting CSIHnCTL0.CSIHnJOBE, reception is forcibly stopped when job 3 ends.
9. After communication is forcibly stopped, the interrupt requests CSIHnTIJC and CSIHnTIR are generated when job 3 ends.

   The interrupt request CSIHnTIJC indicates that reception was forcibly stopped when the current job ended.
   Because the interrupt request CSIHnTIJC is generated instead of the interrupt request CSIHnTIC, the interrupt request CSIHnTIC is not generated.
10. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations.
    In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

25.5.3 Procedures in dual buffer mode

(1) For transmission/reception in the master mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Dual buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 01_B)
- Number of data packets: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

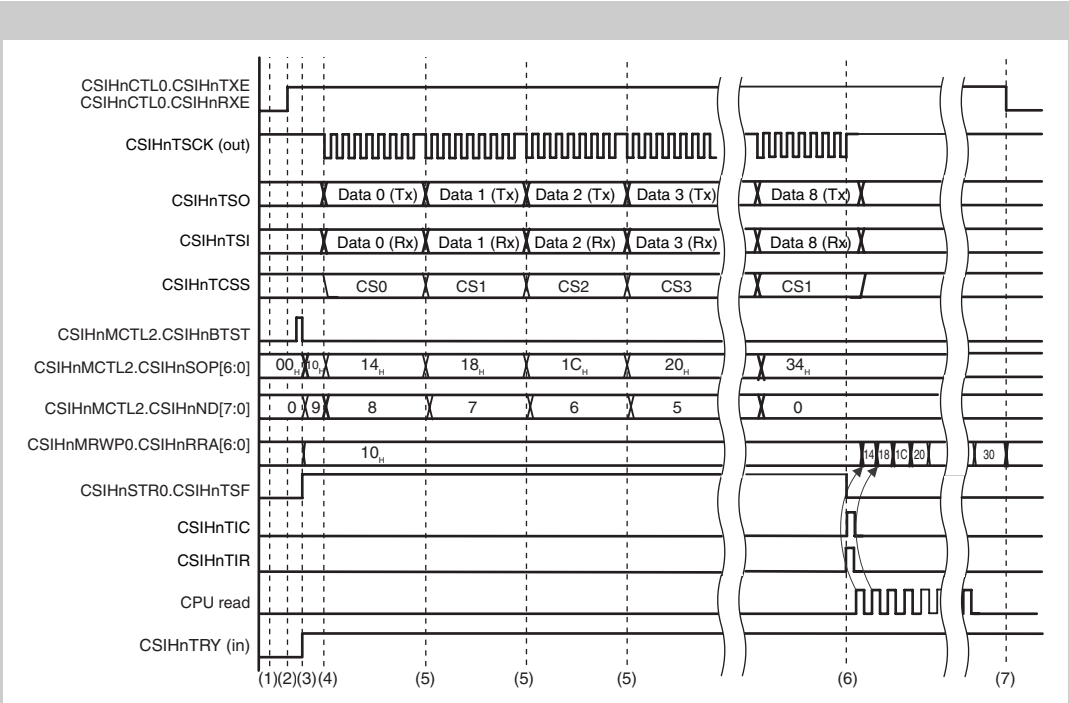


Figure 25-56 Dual buffer mode (for transmission/reception in the master mode, and when the job mode is disabled)

Note The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 01_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS3 are used.)
   CSIHnSTCR0.CSIHnPCT = 1 (buffer pointers cleared)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits. Communication is started by setting CSIHnMCTL2.CSIHnBTST.
4. Transmission starts. Each time a data item is transmitted, the CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented.
5. (4) is repeated until the last data is transmitted/received.
   The interrupt requests CSIHnTIC and CSIHnTIR are not generated.
6. When all the communication ends, the interrupt requests CSIHnTIC and CSIHnTIR are generated.
   The CPU starts reading the received data from the reception buffer. The read start address is specified by the CSIHnMRWP0.CSIHnRRA[6:0] bits. The CSIHnMRWP0.CSIHnRRA[6:0] bits are incremented each time a data item is read.
7. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(2) For reception in the master mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Dual buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 01_B)
- Number of data packets: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

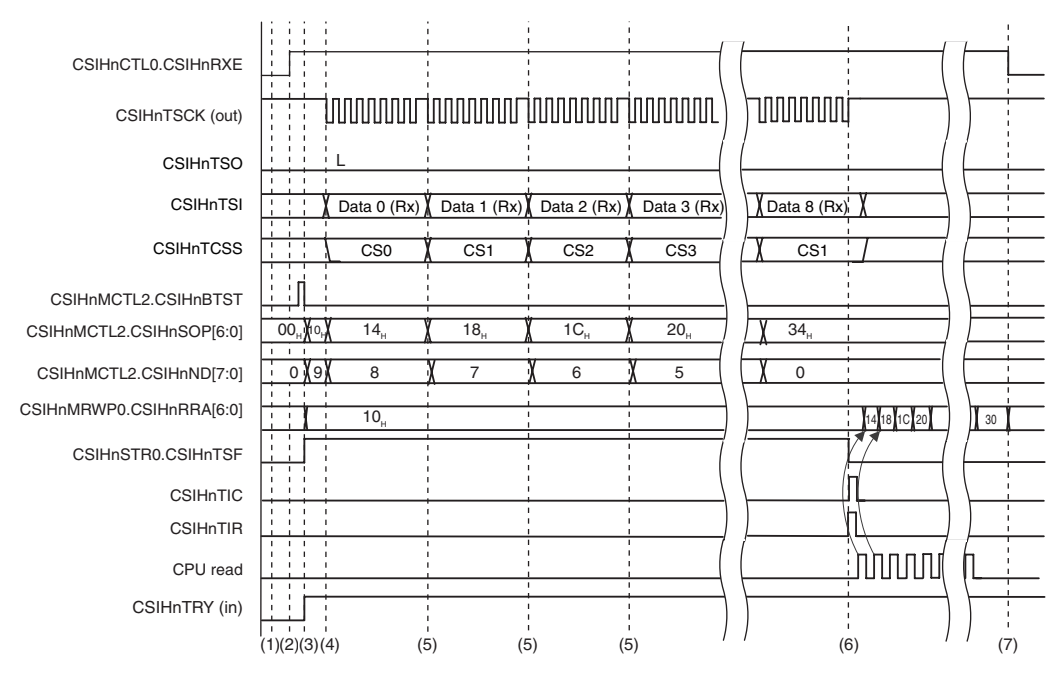


Figure 25-57 Dual buffer mode (for reception in the master mode, and when the job mode is disabled)

Note The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 01_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS3 are used.)
   CSIHnSTCR0.CSIHnPCT = 1 (buffer pointers cleared)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits.
   Reception is started by setting CSIHnMCTL2.CSIHnBTST.
4. Reception starts. Each time a data item is received, the CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented.
5. (4) is repeated until the last data is received.
   The interrupt requests CSIHnTIC and CSIHnTIR are not generated.
6. When all the reception ends, the interrupt requests CSIHnTIC and CSIHnTIR are generated.
   The CPU starts reading the received data from the reception buffer. The read start address is specified by the CSIHnMRWP0.CSIHnRRA[6:0] bits. The CSIHnMRWP0.CSIHnRRA[6:0] bits are incremented each time a data item is read.
7. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations.
   In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(3) For transmission/reception in the slave mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFG0.CSIHnCKP0 = 0, CSIHnCFG0.CSIHnDAP0 = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Handshake enabled (CSIHnCTL1.CSIHnHSE = 1)
- Dual buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 01_B)
- Number of data packets: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

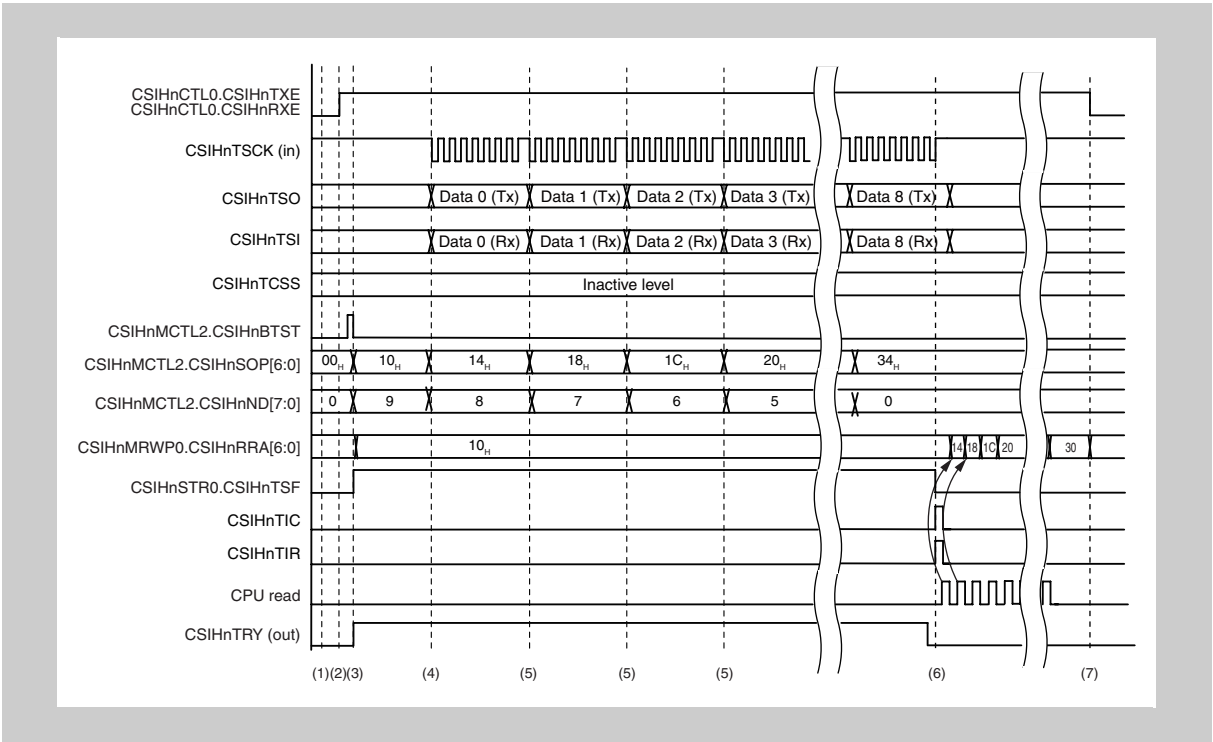


Figure 25-58 Dual buffer mode (for transmission/reception in the slave mode, and when the job mode is disabled)

Note The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 01_B (memory mode)
   CSIHnCFG0 (communication protocol)
   CSIHnSTCR0.CSIHnPCT = 1 (buffer pointers cleared)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits.
   Communication is started by setting CSIHnMCTL2.CSIHnBTST.
4. Transmission starts. Each time a data item is transmitted, the CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented.
5. (4) is repeated until the last data is transmitted/received.
   The interrupt requests CSIHnTIC and CSIHnTIR are not generated.
6. When all the communication ends, the interrupt requests CSIHnTIC and CSIHnTIR are generated.
   The CPU starts reading the received data from the reception buffer. The read start address is specified by the CSIHnMRWP0.CSIHnRRA[6:0] bits. The CSIHnMRWP0.CSIHnRRA[6:0] bits are incremented each time a data item is read.
7. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(4) For reception in the slave mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFG0.CSIHnCKP0 = 0, CSIHnCFG0.CSIHnDAP0 = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Handshake enabled (CSIHnCTL1.CSIHnHSE = 1)
- Dual buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 01_B)
- Number of data packets: 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- Transfer start address: 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

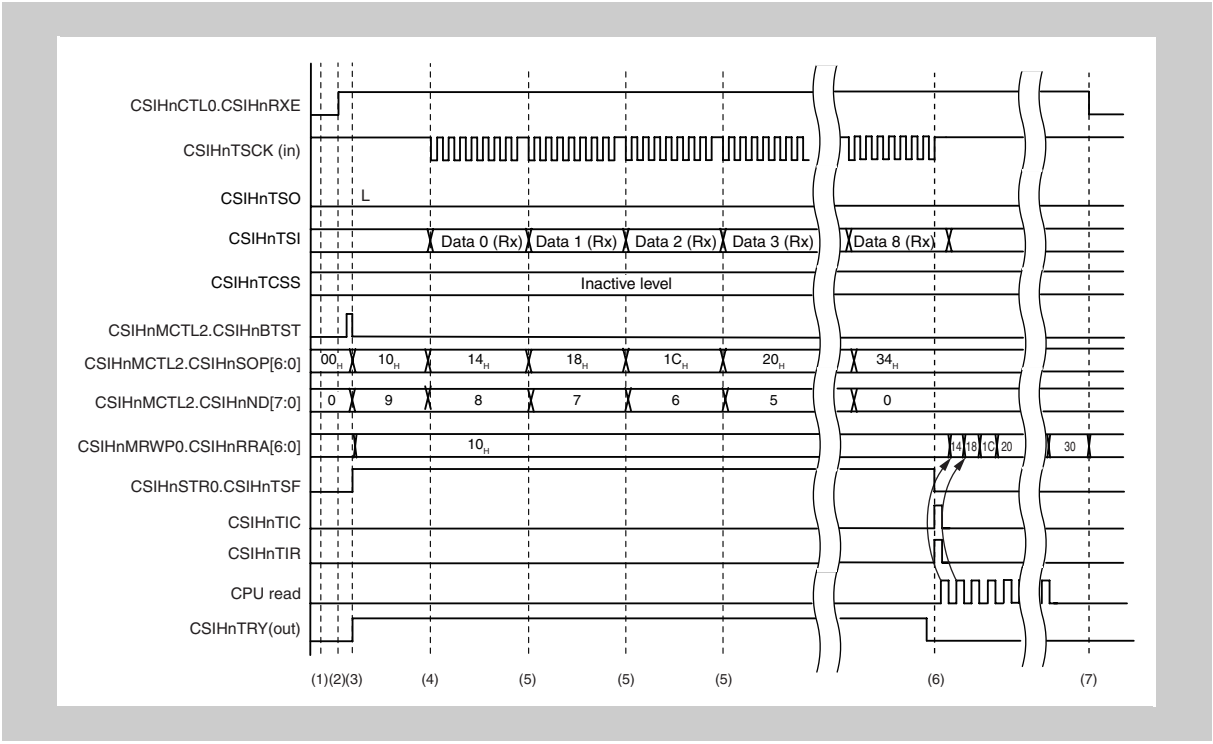


Figure 25-59 Dual buffer mode (for reception in the slave mode, and when the job mode is disabled)

Note The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 01_B (memory mode)
   CSIHnCFG0 (communication protocol)
   CSIHnSTCR0.CSIHnPCT = 1 (buffer pointers cleared)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits.
   Reception is started by setting CSIHnMCTL2.CSIHnBTST.
4. Reception starts. Each time a data item is received, the CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented.
5. (4) is repeated until the last data is received.
   The interrupt requests CSIHnTIC and CSIHnTIR are not generated.
6. When all the reception ends, the interrupt requests CSIHnTIC and CSIHnTIR are generated.
   The CPU starts reading the received data from the reception buffer. The read start address is specified by the CSIHnMRWP0.CSIHnRRA[6:0] bits. The CSIHnMRWP0.CSIHnRRA[6:0] bits are incremented each time a data item is read.
7. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations.
   In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(5) For transmission/reception in the master mode, and when the job mode is enabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- Dual buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 01_B)
- Number of data packets: 12 (CSIHnMCTL2.CSIHnND[7:0] = 12_H)
- Transfer start address: 00_H (CSIHnMCTL2.CSIHnSOP[6:0] = 00_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

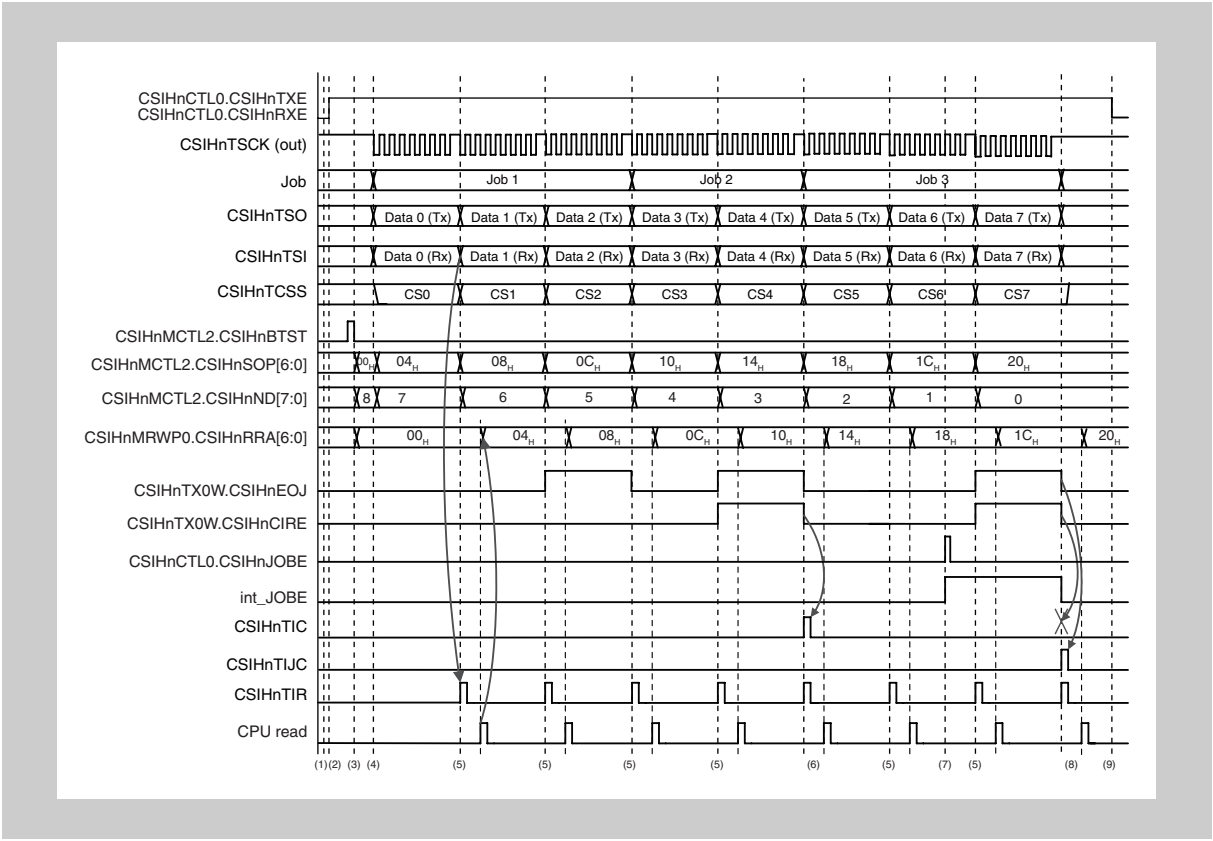


Figure 25-60 Dual buffer mode (for transmission/reception in the master mode, and when the job mode is enabled)

Notes

1. The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.
2. The int_JOBE signal in the above timing chart is the internal signal of the CSIHnCTL0.CSIHnJOBE bit.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 01_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS7 are used.)
   CSIHnSTCR0.CSIHnPCT = 1 (buffer pointers cleared)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits. Communication is started by setting CSIHnMCTL2.CSIHnBTST.
4. Transmission starts. Each time a data item is transmitted, the CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented.
5. When all the data is received, CSIHnTIR is generated. The CSIHnTIR interrupt indicates that the reception data register CSIHnRX0W must be read.
6. CSIHnTIC is generated by setting CSIHnTX0W.CSIHnCIRE to 1. CSIHnTIC indicates that the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted.
7. By setting CSIHnCTL0.CSIHnJOBE to 1, communication is forcibly stopped when job 3 ends.
8. After communication is forcibly stopped, the interrupt requests CSIHnTIJC and CSIHnTIR are generated when job 3 ends.
   The interrupt request CSIHnTIJC indicates that communication was forcibly stopped when the current job ended.
   Because the interrupt request CSIHnTIJC is generated instead of the interrupt request CSIHnTIC, the interrupt request CSIHnTIC is not generated. Note that transfer data is not transmitted by the CSIHnTX0W register.
9. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(6) For reception in the master mode, and when the job mode is enabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- Dual buffer mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 01_B)
- Number of data packets: 12 (CSIHnMCTL2.CSIHnND[7:0] = 12_H)
- Transfer start address: 00_H (CSIHnMCTL2.CSIHnSOP[6:0] = 00_H)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

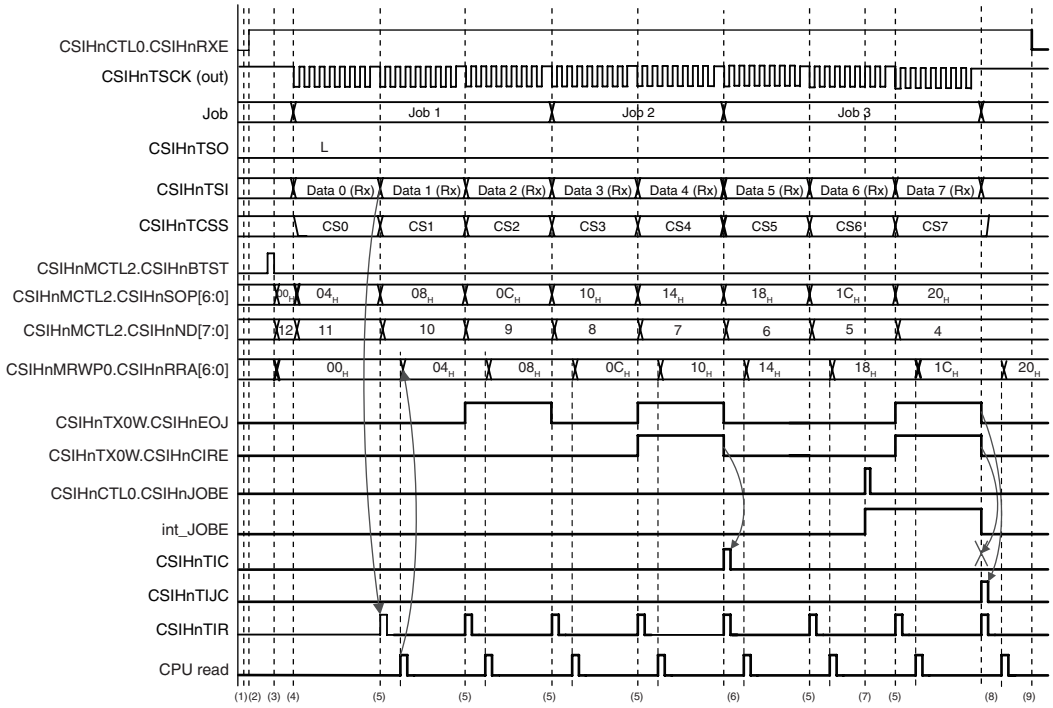


Figure 25-61 Dual buffer mode (for reception in the master mode, and when the job mode is enabled)

Notes

1. The procedure for writing data to the buffer is not described here. The first data address is specified for CSIHnMRWP0.CSIHnTRWA[6:0], and the transfer data is written to CSIHnTX0W. Each time transfer data is written, the value of CSIHnMRWP0.CSIHnTRWA[6:0] is incremented.
2. The int_JOBE signal in the above timing chart is the internal signal of the CSIHnCTL0.CSIHnJOBE bit.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 01_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS7 are used.)
   CSIHnSTCR0.CSIHnPCT = 1 (buffer pointers cleared)
2. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
3. The transmission pointer and number of data items are specified using the CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0] bits. Communication is started by setting CSIHnMCTL2.CSIHnBTST.
4. Reception starts. Each time a data item is received, the CSIHnMCTL2.CSIHnSOP[6:0] bits are automatically incremented, and the CSIHnMCTL2.CSIHnND[7:0] bits are decremented.
5. Each time data is received, CSIHnTIR is generated. The CSIHnTIR interrupt indicates that the reception data register CSIHnRX0W must be read.
6. CSIHnTIC is generated by setting CSIHnTX0W.CSIHnCIRE to 1. CSIHnTIC indicates that the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted.
7. By setting CSIHnCTL0.CSIHnJOBE to 1, reception is forcibly stopped when job 3 ends.
8. After reception is forcibly stopped, the interrupt requests CSIHnTIJC and CSIHnTIR are generated when job 3 ends.
   The interrupt request CSIHnTIJC indicates that reception was forcibly stopped when the current job ended.
   Because the interrupt request CSIHnTIJC is generated instead of the interrupt request CSIHnTIC, the interrupt request CSIHnTIC is not generated. Note that transfer data is not transmitted by the CSIHnTX0W register.
9. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

25.5.4 Procedures in FIFO mode

(1) For transmission/reception in the master mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- FIFO mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 00_B)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

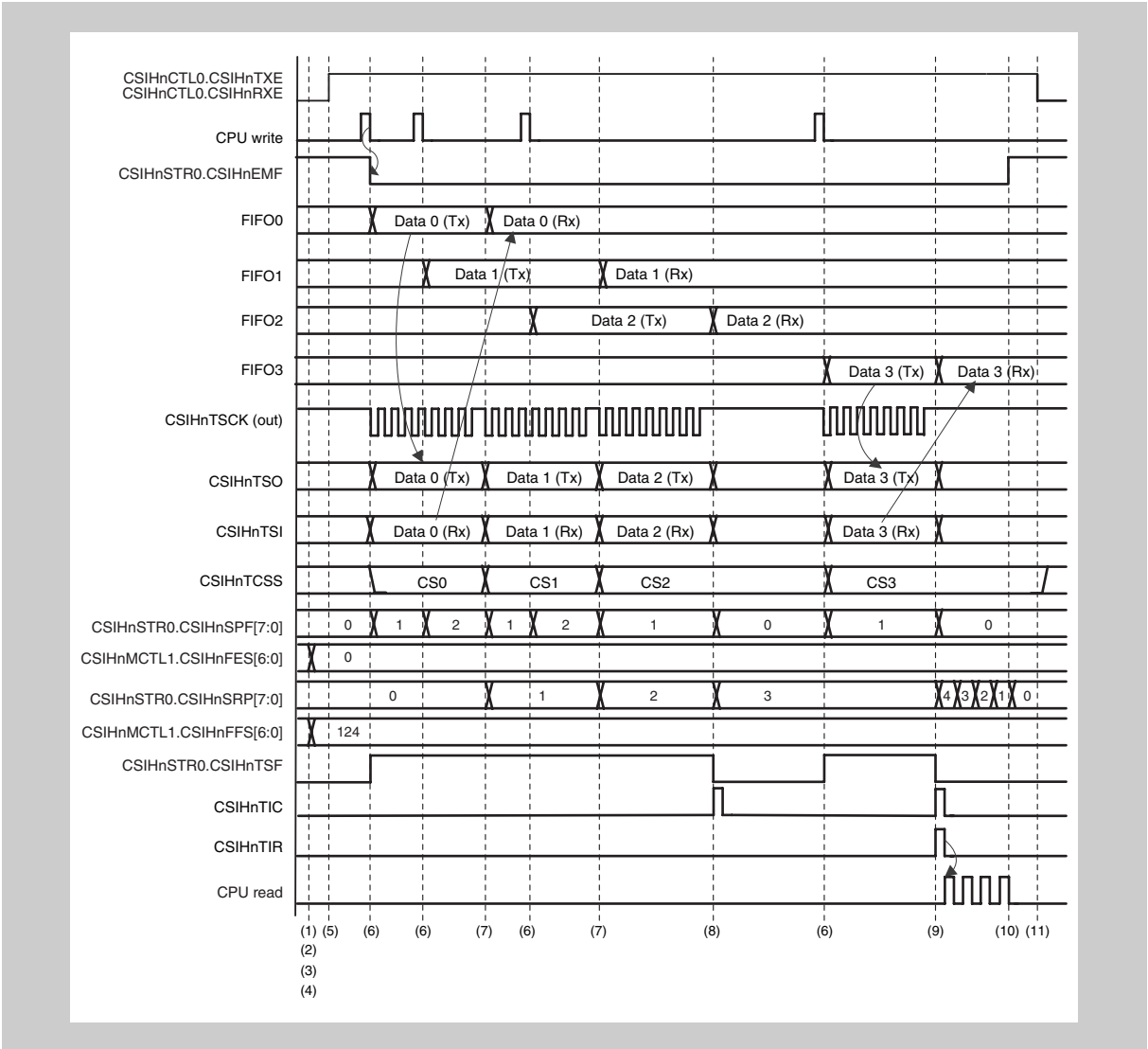


Figure 25-62 FIFO mode (for transmission/reception in the master mode, and when the job mode is disabled)

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 00_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS3 are used.)
2. Set CSIHnSTCR0.CSIHnPCT to 1 to clear all the buffer pointers.
3. Make sure that CSIHnSTR0.CSIHnFLF = 0, CSIHnSTR0.CSIHnEMF = 1, and CSIHnSTR0.CSIHnSPF[7:0] = 00_H.
4. Specify the CSIHnTIC interrupt condition for CSIHnMCTL1.CSIHnFES[6:0].
   Specify the CSIHnTIR interrupt condition for CSIHnMCTL1.CSIHnFFS[6:0].
5. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
6. When transmission data is written to the transmission data register CSIHnTX0W, communication starts.
7. Some of the communication finishes, but CSIHnTIC is not generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] do not match.
8. CSIHnTIC is generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] match.
9. The interrupt request CSIHnTIR is generated because the values of CSIHnMCTL1.CSIHnFFS[6:0] and (128 – CSIHnSTR0.CSIHnSRP[7:0]) match.
   The interrupt request CSIHnTIC is generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] match.
   The CPU starts reading the received data stored in the reception buffer.
10. The CPU finishes reading the received data. The CSIHnSTR0.CSIHnEMF bit is set because the FIFO buffer is empty.
11. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(2) For reception in the master mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- FIFO mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 00_B)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

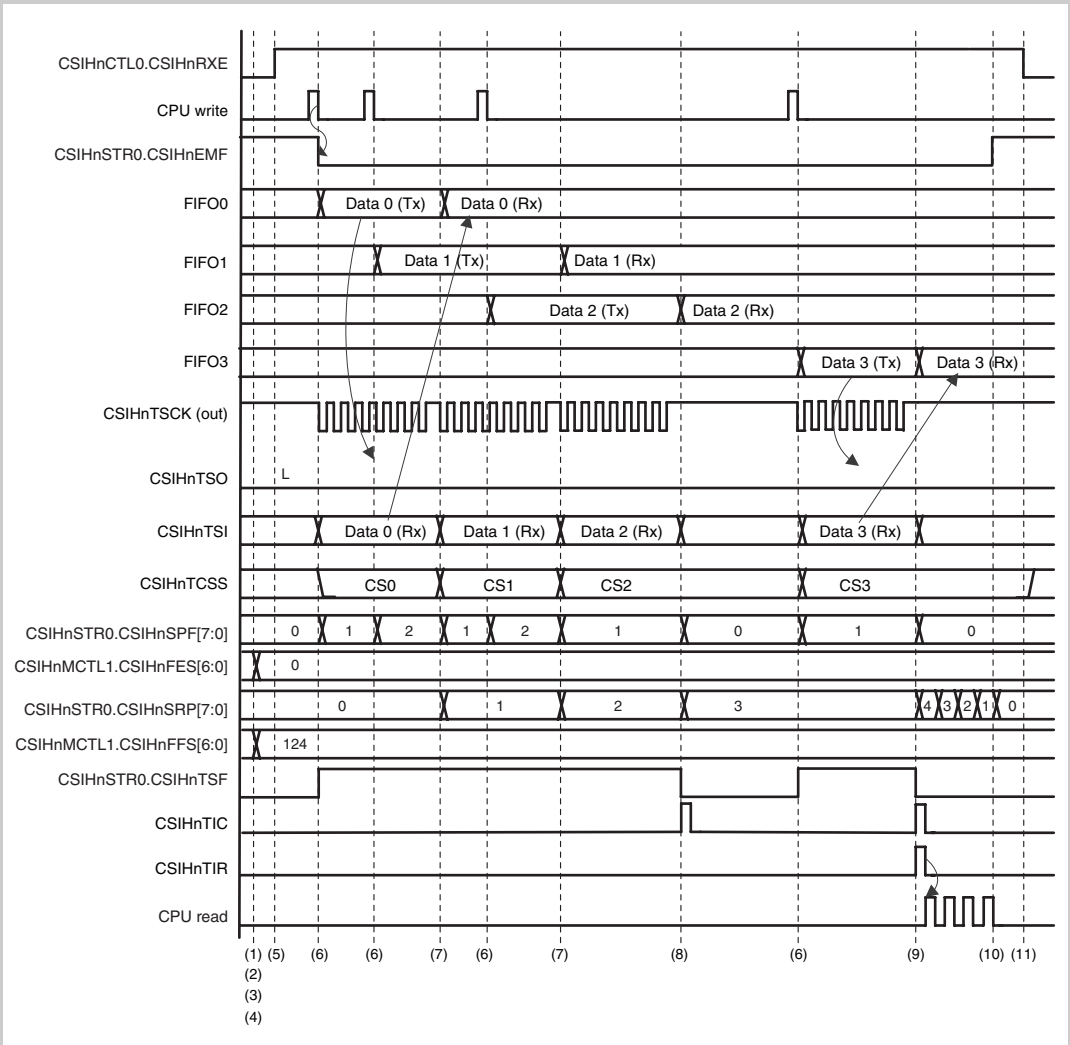


Figure 25-63 FIFO mode (for reception in the master mode, and when the job mode is disabled)

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 00_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS3 are used.)
2. Set CSIHnSTCR0.CSIHnPCT to 1 to clear all the buffer pointers.
3. Make sure that CSIHnSTR0.CSIHnFLF = 0, CSIHnSTR0.CSIHnEMF = 1, and CSIHnSTR0.CSIHnSPF[7:0] = 00_H.
4. Specify the CSIHnTIC interrupt condition for CSIHnMCTL1.CSIHnFES[6:0].
   Specify the CSIHnTIR interrupt condition for CSIHnMCTL1.CSIHnFFS[6:0].
5. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
6. When transmission data is written to the transmission data register CSIHnTX0W, communication starts. (The transmission data is not used, but the chip select signal is enabled.)
7. Some of the communication finishes, but CSIHnTIC is not generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] do not match.
8. CSIHnTIC is generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] match.
9. The interrupt request CSIHnTIR is generated because the values of CSIHnMCTL1.CSIHnFFS[6:0] and (128 – CSIHnSTR0.CSIHnSRP[7:0]) match.
   The interrupt request CSIHnTIC is generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] match.
   The CPU starts reading the received data stored in the reception buffer.
10. The CPU finishes reading the received data. The CSIHnSTR0.CSIHnEMF bit is set because the FIFO buffer is empty.
11. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations.
    In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(3) For transmission/reception in the slave mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFG0.CSIHnDLS0[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFG0.CSIHnDIR0 = 0)
- Normal clock phase and data phase (CSIHnCFG0.CSIHnCKP0 = 0, CSIHnCFG0.CSIHnDAP0 = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Handshake enabled (CSIHnCTL1.CSIHnHSE = 1)
- FIFO mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 00_B)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

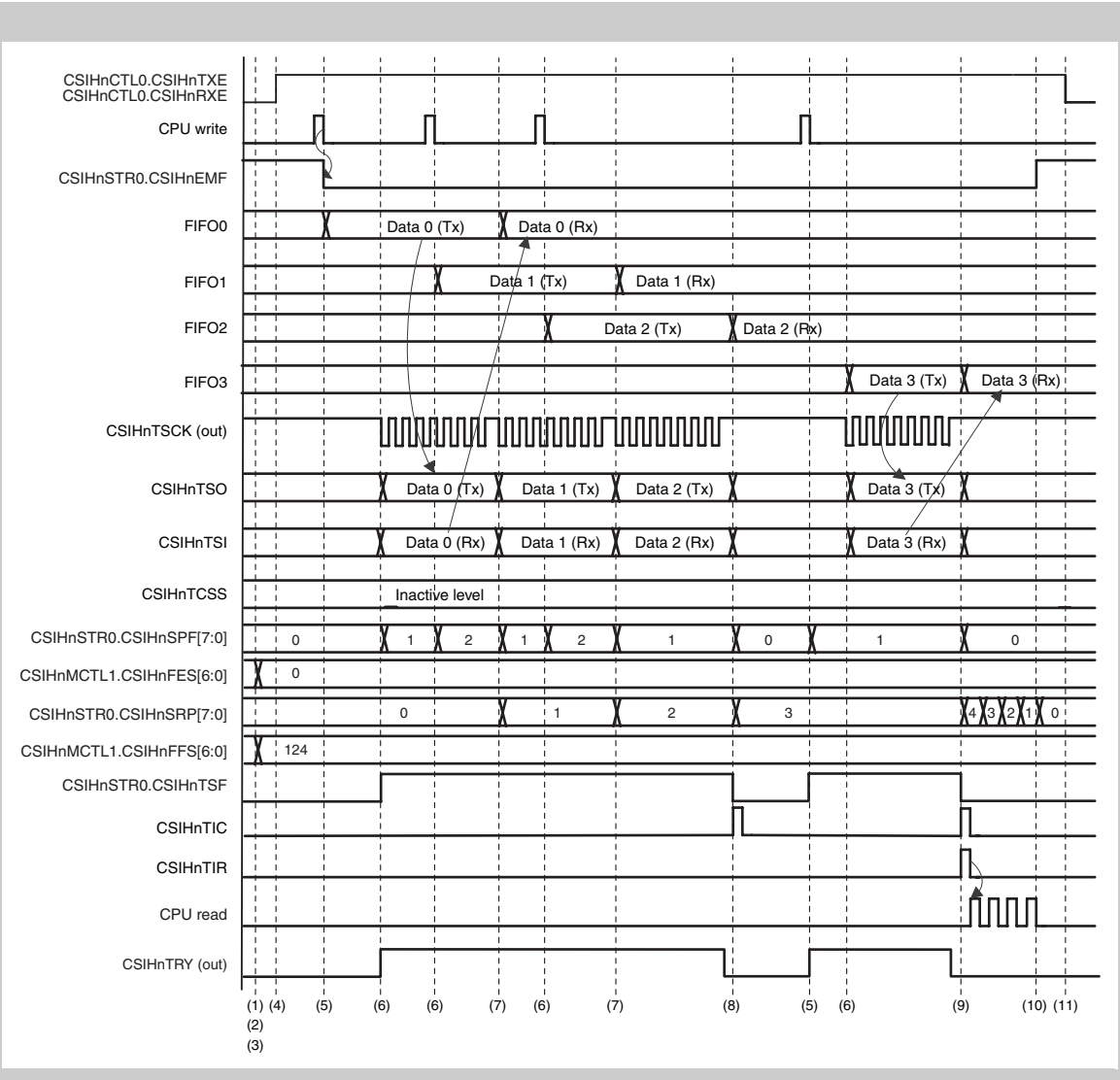


Figure 25-64 FIFO mode (for transmission/reception in the slave mode, and when the job mode is disabled)

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 00_B (memory mode)
   CSIHnCFG0 (communication protocol)
2. Set CSIHnSTCR0.CSIHnPCT to 1 to clear all the buffer pointers.
   Make sure that CSIHnSTR0.CSIHnFLF = 0, CSIHnSTR0.CSIHnEMF = 1, and CSIHnSTR0.CSIHnSPF[7:0] = 00_H.
3. Specify the CSIHnTIC interrupt condition for CSIHnMCTL1.CSIHnFES[6:0].
   Specify the CSIHnTIR interrupt condition for CSIHnMCTL1.CSIHnFFS[6:0].
4. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
5. Write the transfer data to the transmission data register CSIHnTX0W. The CSIHnTRY signal is switched from BUSY (low level) to READY (high level) when data is written.
6. When a serial clock is supplied from the master, communication automatically starts.
7. Some of the communication finishes, but CSIHnTIC is not generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] do not match.
8. CSIHnTIC is generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] match.
9. The interrupt request CSIHnTIR is generated because the values of CSIHnMCTL1.CSIHnFFS[6:0] and (128 – CSIHnSTR0.CSIHnSRP[7:0]) match.
   The interrupt request CSIHnTIC is generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] match.
   The CPU starts reading the received data stored in the reception buffer.
10. The CPU finishes reading the received data. The CSIHnSTR0.CSIHnEMF bit is set because the FIFO buffer is empty.
11. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(4) For reception in the slave mode, and when the job mode is disabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFG0.CSIHnDLS0[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFG0.CSIHnDIR0 = 0)
- Normal clock phase and data phase (CSIHnCFG0.CSIHnCKP0 = 0, CSIHnCFG0.CSIHnDAP0 = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- Handshake enabled (CSIHnCTL1.CSIHnHSE = 1)
- FIFO mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 00_B)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

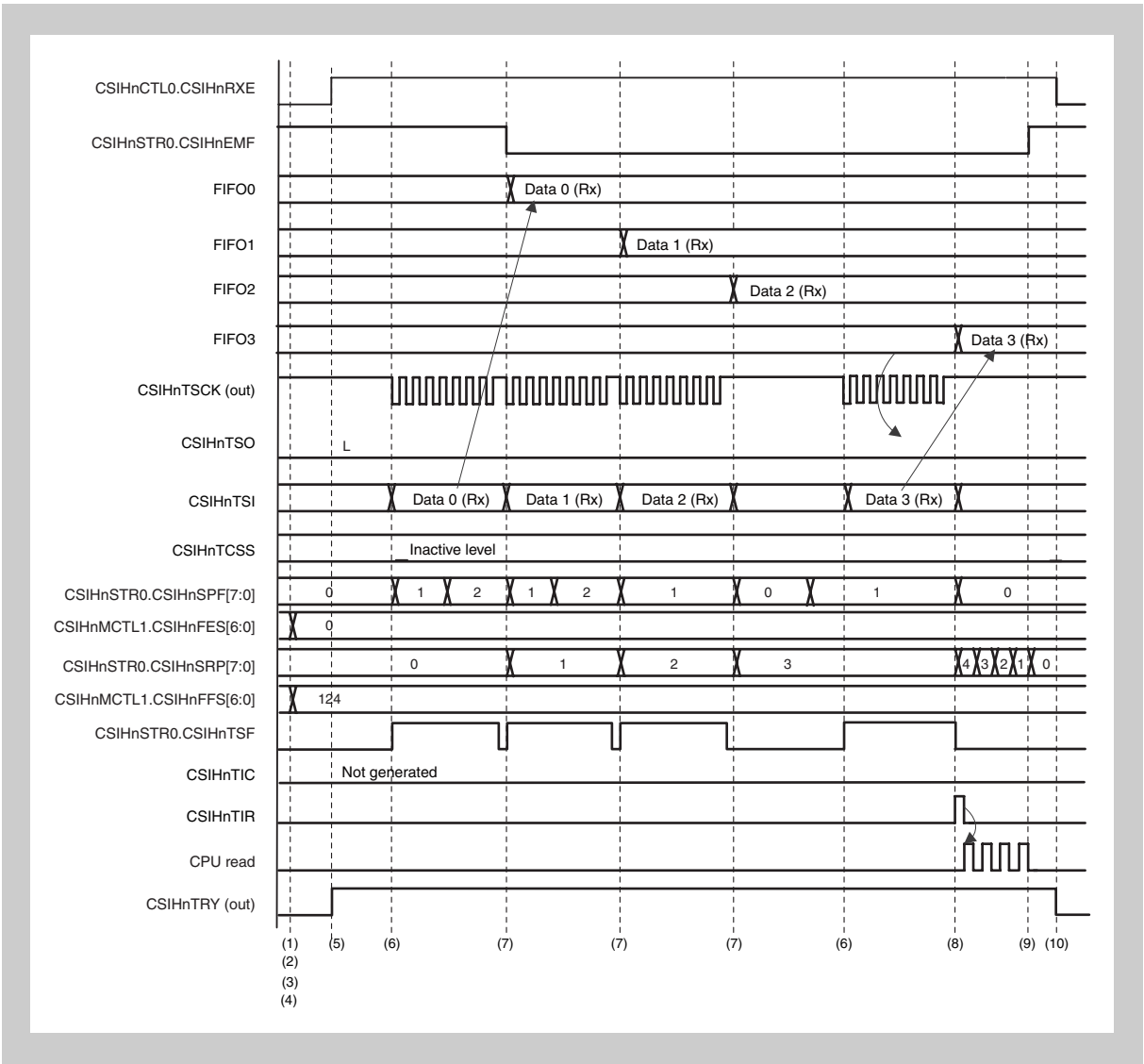


Figure 25-65 FIFO mode (for reception in the slave mode, and when the job mode is disabled)

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 00_B (memory mode)
   CSIHnCFG0 (communication protocol)
2. Set CSIHnSTCR0.CSIHnPCT to 1 to clear all the buffer pointers.
3. Make sure that CSIHnSTR0.CSIHnFLF = 0, CSIHnSTR0.CSIHnEMF = 1, and CSIHnSTR0.CSIHnSPF[7:0] = 00_H.
4. Specify the CSIHnTIR interrupt condition for CSIHnMCTL1.CSIHnFFS[6:0].
5. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 0 (transmission disabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
   The CSIHnTRY signal is switched from BUSY (low level) to READY (high level) by clearing CSIHnCTL0.CSIHnPWR to 0 and setting CSIHnCTL0.CSIHnRXE to 1.
6. When a serial clock is supplied from the master, reception automatically starts.
7. Some of the communication finishes, but CSIHnTIC is not generated because the system is in the reception mode.
8. The interrupt request CSIHnTIR is generated because the values of CSIHnMCTL1.CSIHnFFS[6:0] and (128 – CSIHnSTR0.CSIHnSRP[7:0]) match.
   The CPU starts reading the received data stored in the reception buffer.
9. The CPU finishes reading the received data. The CSIHnSTR0.CSIHnEMF bit is set because the FIFO buffer is empty.
10. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations.
    In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(5) For transmission/reception in the master mode, and when the job mode is enabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode is enabled (CSIHnCTL1.CSIHnJE = 1)
- FIFO mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 00_B)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)
- Job 1 = four data items, job 2 = three data items, and job 3 = five data items

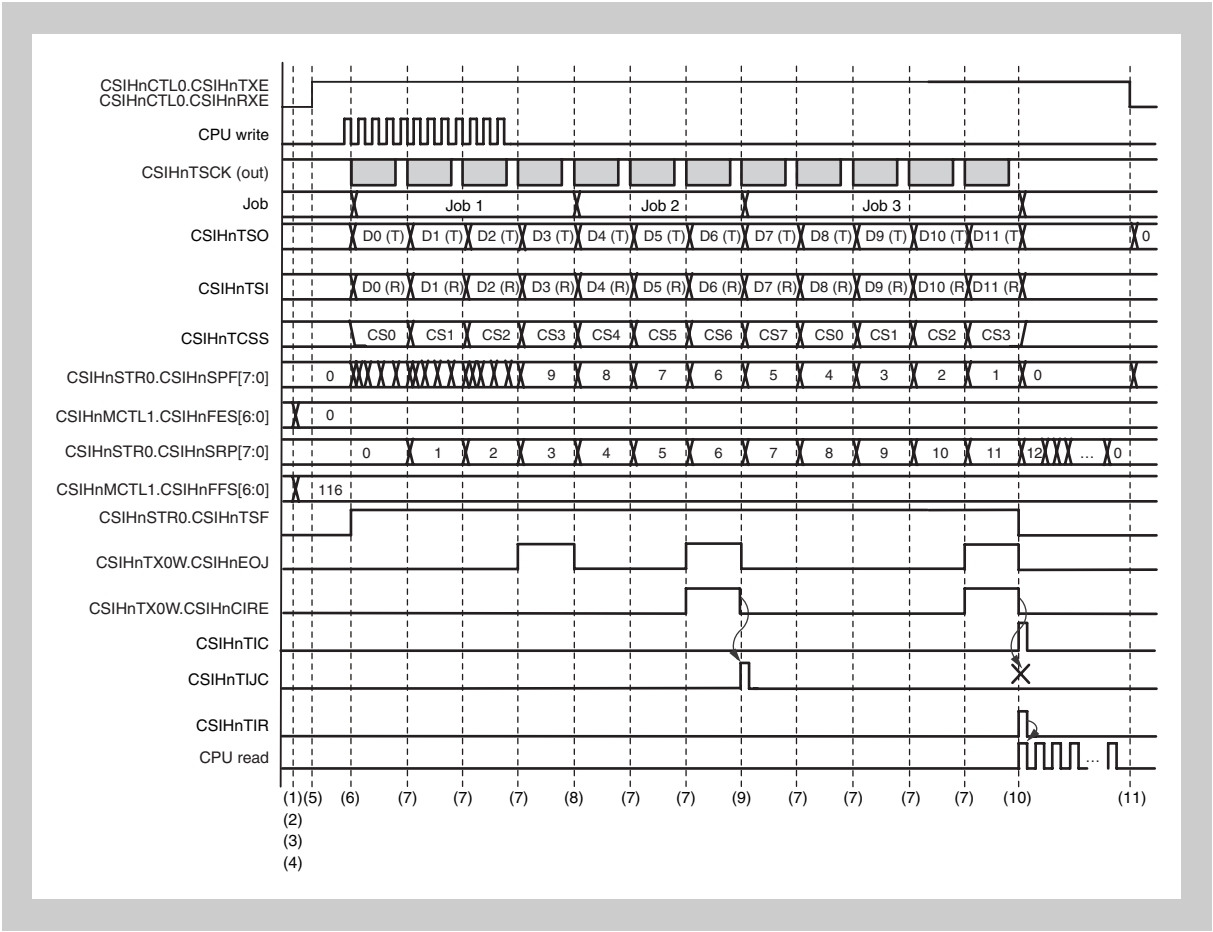


Figure 25-66 FIFO mode (for transmission/reception in the master mode, and when the job mode is enabled)

Note The int_JOBE signal in the above timing chart is the internal signal of the CSIHnCTL0.CSIHnJOBE bit.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 00_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS7 are used.)
2. Set CSIHnSTCR0.CSIHnPCT to 1 to clear all the buffer pointers.
3. Make sure that CSIHnSTR0.CSIHnFLF = 0, CSIHnSTR0.CSIHnEMF = 1, and CSIHnSTR0.CSIHnSPF[7:0] = 00_H.
4. Specify the CSIHnTIC interrupt condition for CSIHnMCTL1.CSIHnFES[6:0].
   Specify the CSIHnTIR interrupt condition for CSIHnMCTL1.CSIHnFFS[6:0].
5. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
6. When transmission data is written to the transmission data register CSIHnTX0W, communication starts.
7. Some of the communication finishes, but CSIHnTIC is not generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] do not match.
8. Because the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted by clearing CSIHnTX0W.CSIHnCIRE, the interrupt request CSIHnTIC is not generated.
9. Because the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted by setting CSIHnTX0W.CSIHnCIRE, the interrupt request CSIHnTIC is generated.
10. CSIHnTIC is generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] match.
    Because CSIHnTIC was generated, CSIHnTIJC is not generated.
    The interrupt request CSIHnTIR is generated because the values of CSIHnMCTL1.CSIHnFFS[6:0] and (128 – CSIHnSTR0.CSIHnSRP[7:0]) match.
    The CPU starts reading the received data stored in the reception buffer.
11. Finally, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE to disable transmission/reception operations. In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

(6) For reception in the master mode, and when the job mode is enabled

The following conditions are assumed for the procedure shown here:

- Transmission data length: 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Transmission direction: MSB first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock phase and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No delay for any interrupt (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- FIFO mode (CSIHnCTL0.CSIHnMBS = 0, CSIHnMCTL0.CSIHnMMS[1:0] = 00_B)
- Normal CSIHnTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)
- Job 1 = four data items, job 2 = three data items, and job 3 = five data items

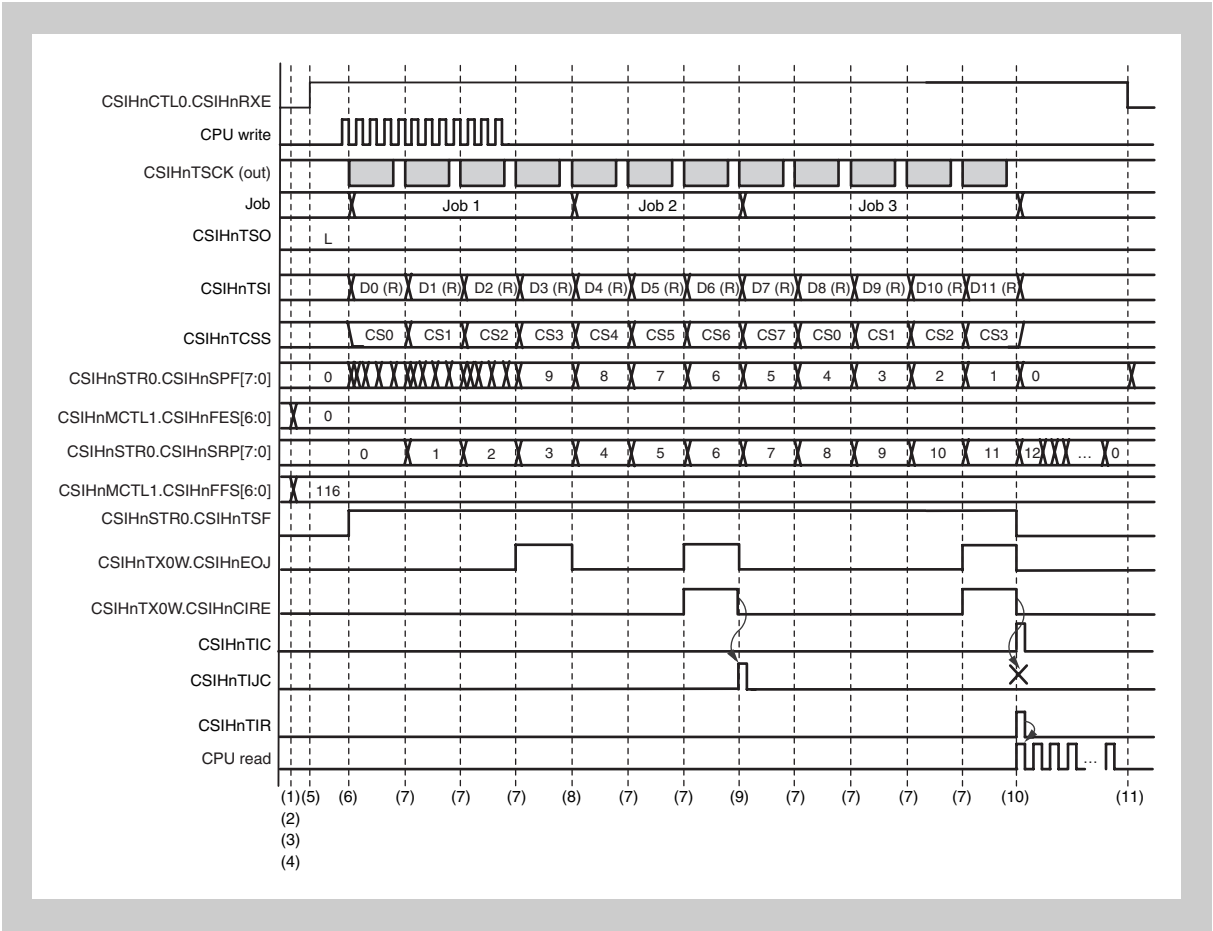


Figure 25-67 FIFO mode (for reception in the master mode, and when the job mode is enabled)

Note The int_JOBE signal in the above timing chart is the internal signal of the CSIHnCTL0.CSIHnJOBE bit.

Procedure:

1. Set up the following registers before setting CSIHnCTL0.CSIHnPWR to 1:
   CSIHnCTL1, CSIHnCTL2 (transfer mode, operating mode)
   CSIHnMCTL0.CSIHnMMS[1:0] = 00_B (memory mode)
   CSIHnCFGx (communication protocol)
   (For this example, the chip select signals CS0 to CS7 are used.)
2. Set CSIHnSTCR0.CSIHnPCT to 1 to clear all the buffer pointers.
3. Make sure that CSIHnSTR0.CSIHnFLF = 0, CSIHnSTR0.CSIHnEMF = 1, and CSIHnSTR0.CSIHnSPF[7:0] = 00_H.
4. Specify the CSIHnTIC interrupt condition for CSIHnMCTL1.CSIHnFES[6:0].
   Specify the CSIHnTIR interrupt condition for CSIHnMCTL1.CSIHnFFS[6:0].
5. CSIHnCTL0.CSIHnPWR = 1 (clock enabled)
   CSIHnCTL0.CSIHnTXE = 1 (transmission enabled)
   CSIHnCTL0.CSIHnRXE = 1 (reception enabled)
   CSIHnCTL0.CSIHnMBS = 0 (memory mode)
6. When transmission data is written to the transmission data register CSIHnTX0W, communication starts. (The transmission data is not used, but the chip select signal is enabled.)
7. Some of the reception finishes, but CSIHnTIC is not generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] do not match.
8. Because the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted by clearing CSIHnTX0W.CSIHnCIRE, the interrupt request CSIHnTIC is not generated.
9. Because the last data (CSIHnTX0W.CSIHnEOJ = 1) of the current job was transmitted by setting CSIHnTX0W.CSIHnCIRE, the interrupt request CSIHnTIC is generated.
10. CSIHnTIC is generated because the values of CSIHnSTR0.CSIHnSPF[7:0] and CSIHnMCTL1.CSIHnFES[6:0] match.
    Because CSIHnTIC was generated, CSIHnTIJC is not generated.
    The interrupt request CSIHnTIR is generated because the values of CSIHnMCTL1.CSIHnFFS[6:0] and (128 – CSIHnSTR0.CSIHnSRP[7:0]) match.
    The CPU starts reading the received data stored in the reception buffer.
11. Finally, clear CSIHnCTL0.CSIHnRXE to disable reception operations.
    In addition, clear CSIHnCTL0.CSIHnPWR to reduce the power consumption while not using the CSIH.

Chapter 26 IICB Bus (IICB)

Caution To use the IIC bus function, use the SCLn and SDAn pins, and set them to N-ch open drain.

26.1 V850E2/Sx4-H IICB Features

Instances This microcontroller has the following number of instances of the IICB:

Table 26-1 Instances of IICB

IICB	
Number of instances	4
Names	IICB0 to IICB3

Instances index n Throughout this chapter, the individual instances of the IICB are identified by the index “n” (n = 0 to 3), for example, IICBnDAT for the IICBn data register.

Register addresses All IICBn register addresses are given as addresses offset from the individual base addresses <IICBn_base>.
The base address <IICBn_base> of each IICBn is listed in the following table:

Table 26-2 Register base addresses <IICBn_base>

IICBn	<IICBn_base> address
IICB0	FF82 0000H
IICB1	FF82 1000H
IICB2	FF82 2000H
IICB3	FF82 3000H

Clock supply The following clock is supplied to IICBn:

Table 26-3 IICBn clock supply

IICBn	Clock	Connected to:
IICB0	PCLK	Clock generator CKSCLK_108
IICB1	PCLK	Clock generator CKSCLK_108
IICB2	PCLK	Clock generator CKSCLK_107
IICB3	PCLK	Clock generator CKSCLK_107

Interrupts and DMA

IICB can generate following interrupt and DMA requests:

Table 26-4 IICBn interrupt and DMA requests

IISAn signals	Function	Connected to:
IICB0:		
IICBTIA0	Data interrupt request signal	Interrupt controller INTIICB0IA DMA controller trigger 57
IICBTIS0	Status interrupt request signal	Interrupt controller INTIICB0IS
IICB1:		
IICBTIA1	Data interrupt request signal	Interrupt controller INTIICB1IA DMA controller trigger 58
IICBTIS1	Status interrupt request signal	Interrupt controller INTIICB1IS
IICB2:		
IICBTIA2	Data interrupt request signal	Interrupt controller INTIICB2IA DMA controller trigger 74
IICBTIS2	Status interrupt request signal	Interrupt controller INTIICB2IS
IICB3:		
IICBTIA3	Data interrupt request signal	Interrupt controller INTIICB3IA DMA controller trigger 75
IICBTIS3	Status interrupt request signal	Interrupt controller INTIICB3IS

IICB hardware reset

IICB and its registers are initialized by the following reset signal:

Table 26-5 IICBn reset signal

IICBn	Reset signal
IICBn	System reset SYSRES

Interrupts and DMA

IICB can generate following interrupt and DMA requests:

Table 26-6 IICBn interrupt and DMA requests

IICBn signals	Function	Connected to:
IICB0:		
SCL0	Serial clock I/O	Port IICB0SCL
SDA0	Serial transmission/reception data I/O	Port IICB0SDA
IICB1:		
SCL1	Serial clock I/O	Port IICB1SCL
SDA1	Serial transmission/reception data I/O	Port IICB1SDA
IICB2:		
SCL2	Serial clock I/O	Port IICB2SCL
SDA2	Serial transmission/reception data I/O	Port IICB2SDA
IICB3:		
SCL3	Serial clock I/O	Port IICB3SCL
SDA3	Serial transmission/reception data I/O	Port IICB3SDA

26.2 Functional Overview

Operating mode	Standard mode (SCL clock frequency: 100 kHz max.) Fast mode (SCL clock frequency: 400 kHz max.)
Transfer mode	Single transfer mode Continuous transfer mode
Pin configuration	SCLn: Serial clock pin SDAn: Serial transmit/receive data pin
Interrupt request signal	Data transmit/receive interrupt request signal (IICBTIAn) Status interrupt request signal (IICBTISn)
Communication data length	8 bits
Multimaster support	Multiple masters can control the bus simultaneously.
SCLn level width	The high-level width and low-level width of the serial clock signal (SCLn) can be changed.
Automatic detection	The start and stop conditions can be detected automatically.

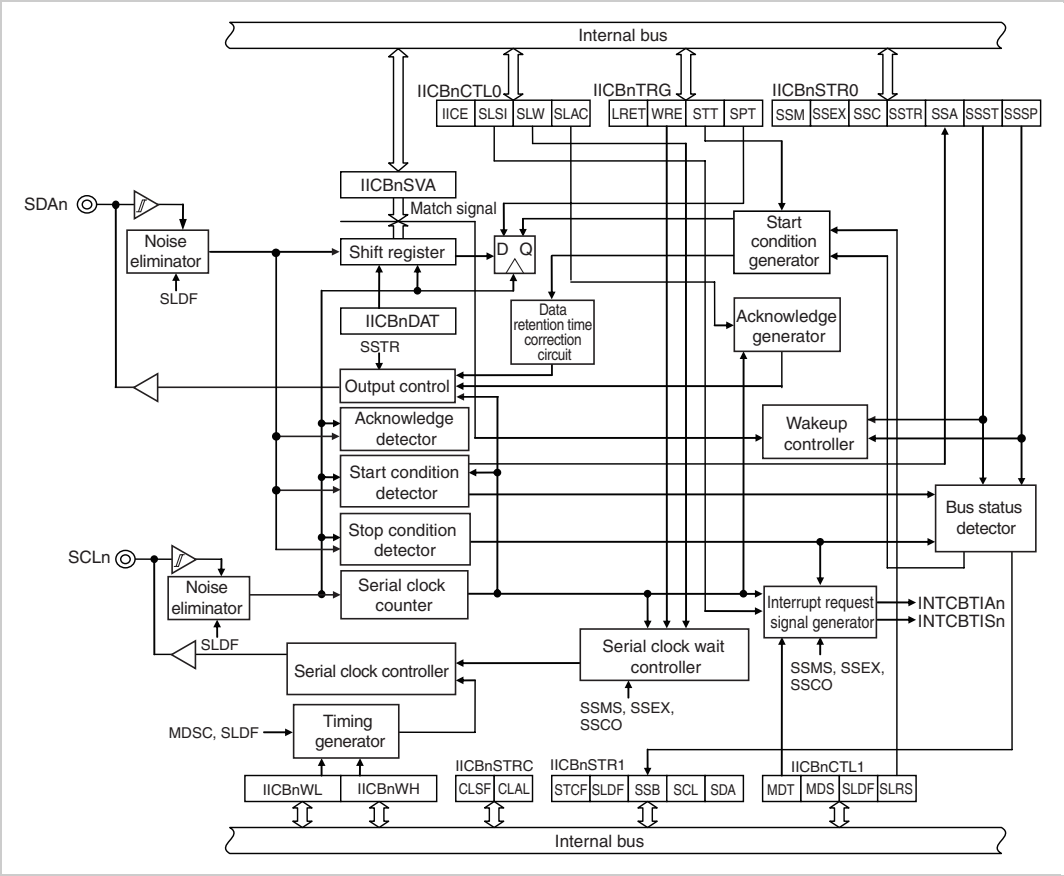


Figure 26-1 Block diagram of IICBn

26.3 IIC Bus Mode Functions

26.3.1 Pin configuration

The serial clock pin (SCLn) and serial data bus pin (SDAn) are configured as follows:

SCLn: This pin is used for serial clock input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

SDAn: This pin is used for serial data input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

Because the outputs of the serial clock line and serial data bus line are N-ch open-drain outputs, an external pull-up resistor must be connected to these lines.

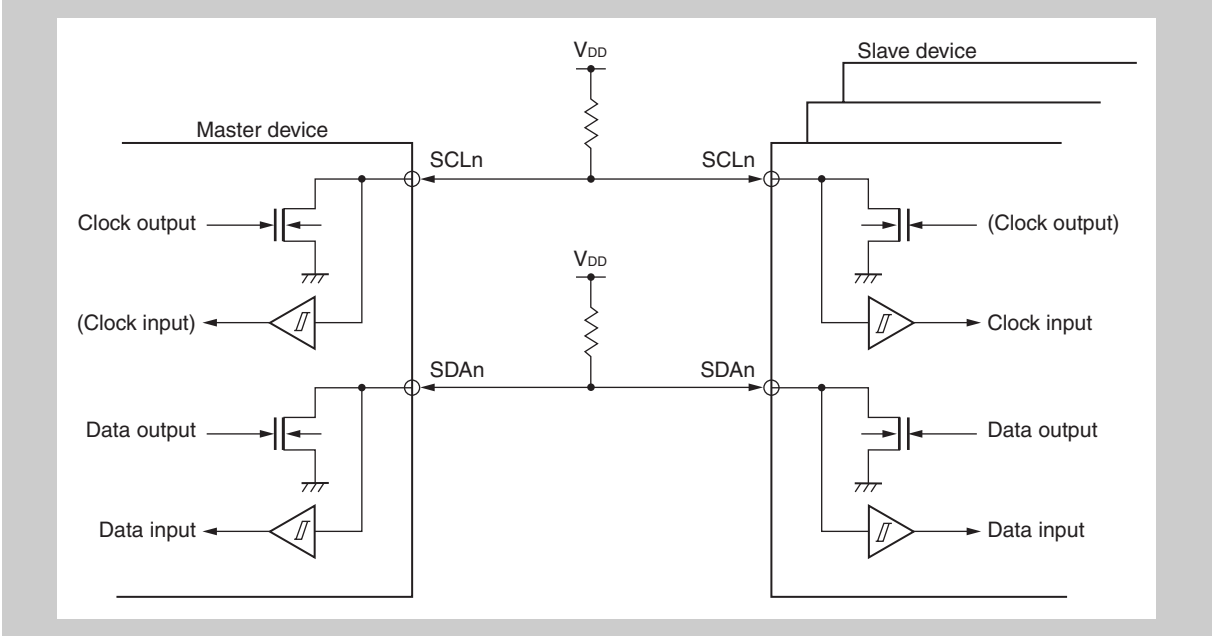


Figure 26-2 Pin configuration diagram

26.4 IIC Bus Definition

This section describes the IIC bus serial data communication format and the signals used by the IIC bus.

Figure 26-3 "IIC bus serial data transfer timing" shows the transfer timing for the start condition, address, transfer direction specification, data, and stop condition, which are output onto the IIC bus serial data bus.

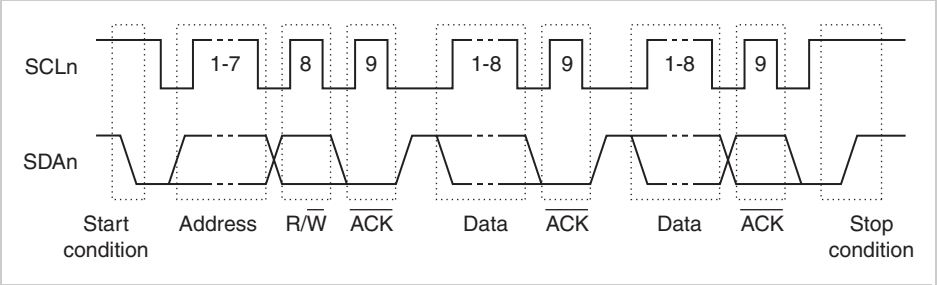


Figure 26-3 IIC bus serial data transfer timing

The start condition, slave address, and stop condition are output by the master device.

$\overline{\text{ACK}}$ can be output by either the master or slave device. (Normally, it is output by the device that receives 8-bit data.)

The serial clock signal (SCLn) is continuously output by the master device. In the slave device, the low-level period of the SCLn signal can be extended to insert a wait.

26.4.1 Start condition

The start condition is met if the SDAn signal level changes from high to low while the SCLn signal is high. The start condition is output when the master device starts serial data transfer to a slave device. When the IICBn is in the slave mode, it detects the start condition.

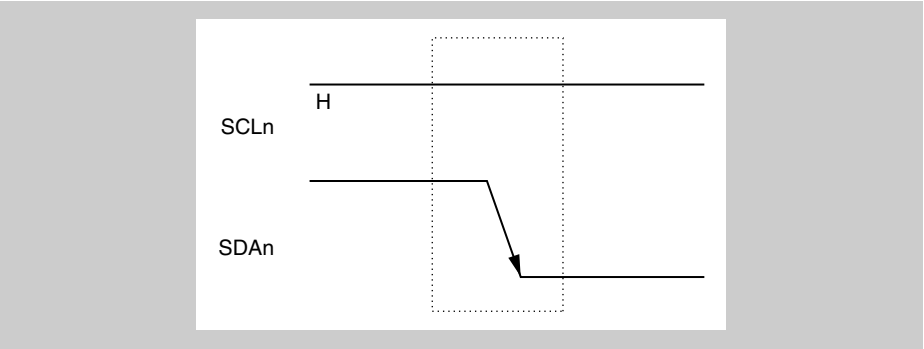


Figure 26-4 Start condition

26.4.2 Addresses

The 7 bits of data following the start condition are defined as an address.

An address is a 7-bit data segment that is output in order to select one of the slave devices that are connected to the master device via the bus lines. Therefore, each slave device connected via the bus lines must have a unique address.

The slave device checks whether the 7-bit data matches its own address. If they match, that slave device is selected as the communication destination and communicates with the master device until the master device outputs another start condition or a stop condition.

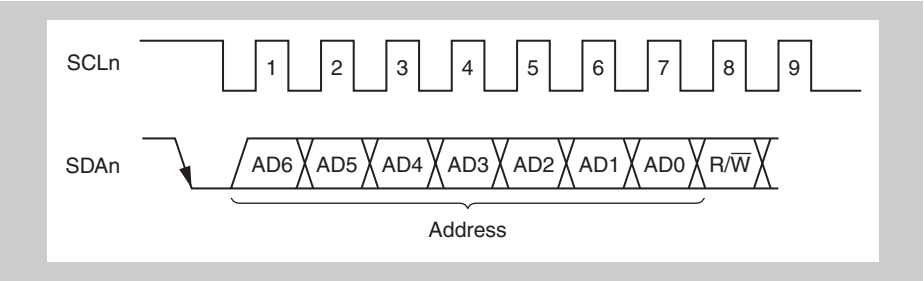


Figure 26-5 Addresses

26.4.3 Extension code

When the higher 4 bits of the address are 0000 or 1111, these bits are called extension code. *Table 26-7 "Extension code bit definitions"* lists the bit definitions of extension code.

Table 26-7 Extension code bit definitions

Slave address	R/W bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	×	CBUS address
0000 010	×	Address reserved for different bus format
0000 011	×	Reserved for future use
0000 1xx	×	HS mode master code[a]
1111 0xx	×	10-bit slave address specification
1111 1xx	×	Reserved for future use

a) The HS mode cannot be used for IICB.

26.4.4 Transfer direction specification

After the 7-bit address data, the master device transmits 1 bit that specifies the transfer direction.

If this transfer direction specification bit is 0, it indicates that the master device transmits data to a slave device. If this bit is 1, it indicates that the master device receives data from a slave device.

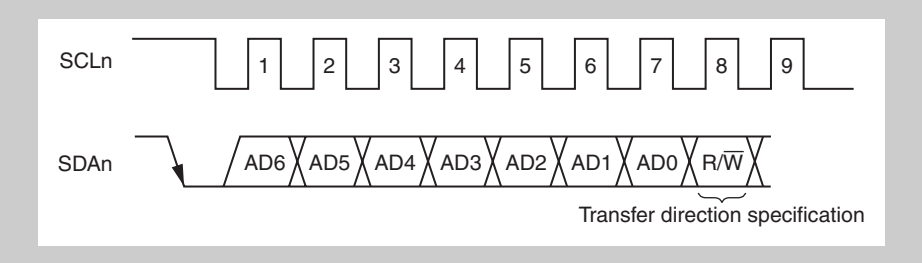


Figure 26-6 Transfer direction specification

26.4.5 Acknowledge ($\overline{ACK}$)

The 1-bit data after the transfer direction bit (R/$\overline{W}$) and the 1-bit data after the 8-bit data during address transfer are defined as an acknowledge signal ($\overline{ACK}$). $\overline{ACK}$ is used to check the serial data status of the transmitting and receiving devices.

The receiving device returns $\overline{ACK}$ after receiving 8-bit data.

The transmitting device normally receives $\overline{ACK}$ after transmitting 8-bit data. If the transmitting device receives $\overline{ACK}$ from the receiving device, it continues processing assuming that the transmitted data is normally received.

If the master device is the receiving device and receives the final data, it does not return $\overline{ACK}$ and outputs a stop condition. If the slave device is the receiving device and does not return $\overline{ACK}$, the master device outputs either a stop condition or a restart condition and then stops the current transmission. Failure to return $\overline{ACK}$ may be caused by the following factors.

1. The transmitted data has not been received normally.
2. The final data has been received.
3. The receiving device (slave) does not exist for the specified address.

$\overline{ACK}$ is output when the SDAn line of the receiving device changes to low level at the 9th clock (normal reception).

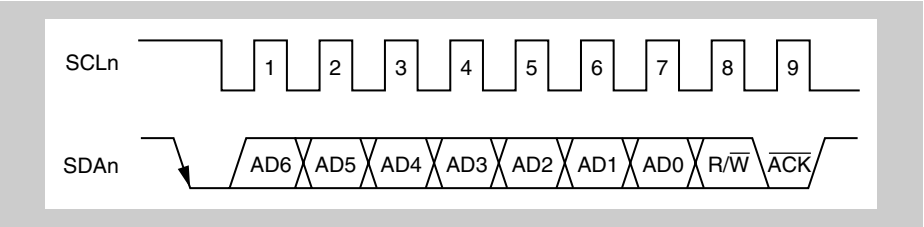


Figure 26-7 Acknowledge ($\overline{ACK}$)

26.4.6 Data

The bits other than the nine bits following the start condition (seven address bits, an R/$\overline{W}$ bit, and an acknowledge ($\overline{ACK}$) bit) and the acknowledge bits are defined as data.

If a 10-bit address is specified using an extension code, the 8-bit data that is transferred after the address is used as the second address.

26.4.7 Stop condition

A stop condition is met if the SDAn signal level changes from low to high while the SCLn signal is high.
The stop condition is output when serial data transfer from the master device to the slave device has been completed.

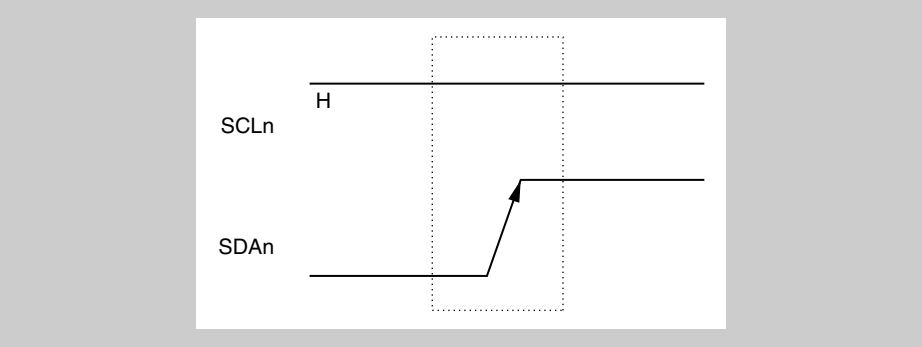


Figure 26-8 Stop condition

26.4.8 Wait state

A wait state is used to report to the communication destination that the IICBn (master or slave) is preparing to transmit or receive data.

The wait state is reported to the communication destination by setting the SCLn signal to low. The next data transfer cannot start until both the master and slave devices exit the wait state.

(a) When a wait at the 9th clock is specified for the master and a wait at the 8th clock for the slave (master: transmission, slave: reception)

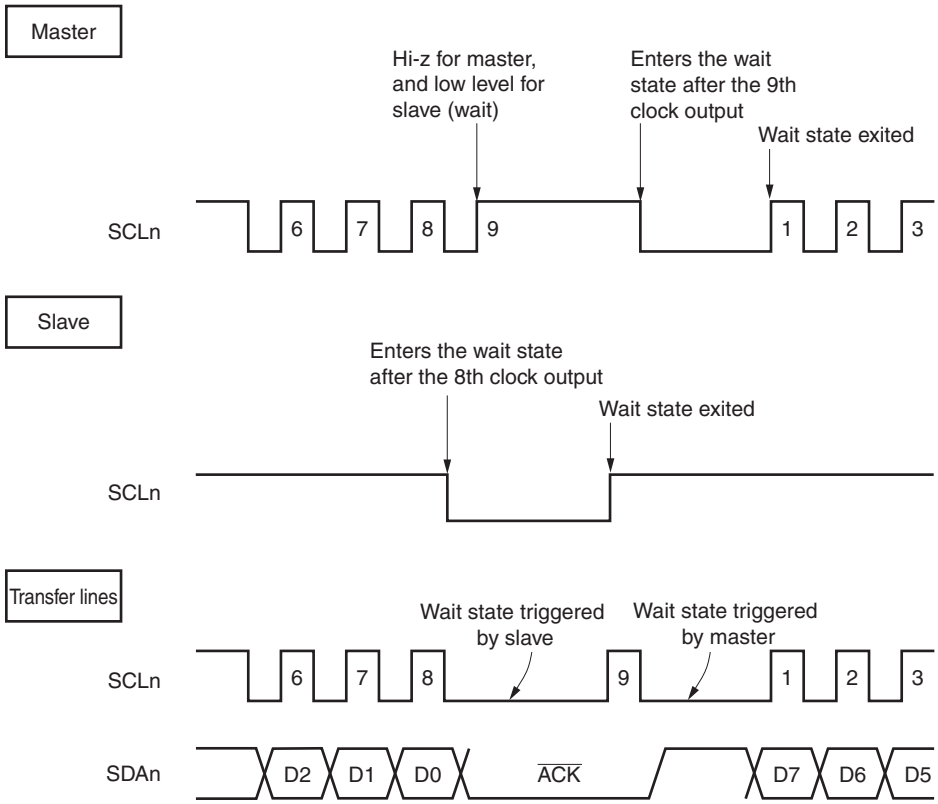


Figure 26-9 Wait state (1/2)

(b) When a wait at the 9th clock is specified for both the master and slave
(master: transmission, slave: reception)

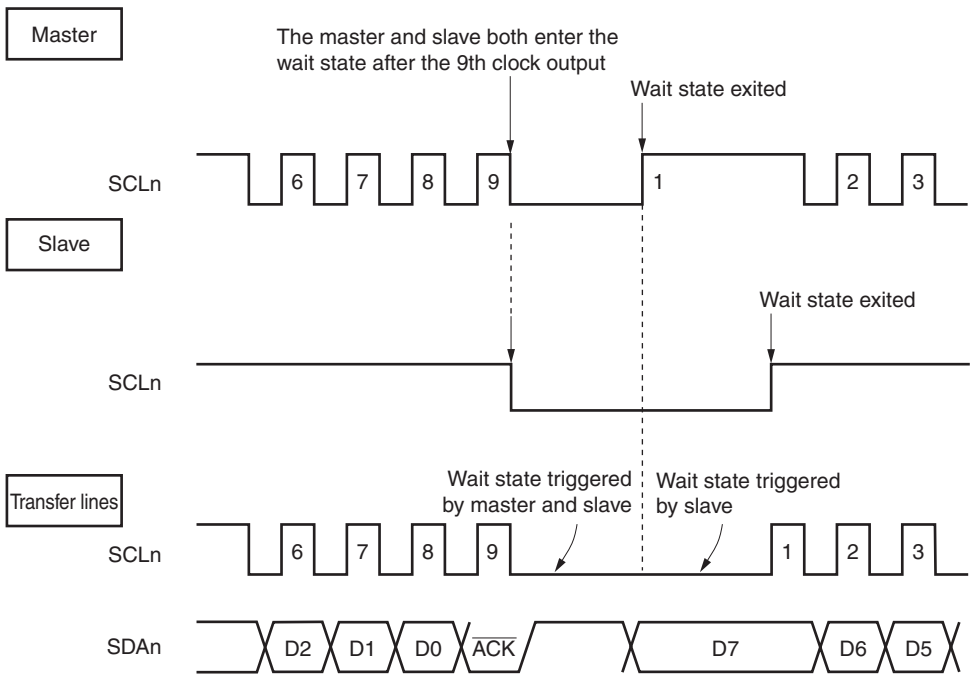


Figure 26-9 Wait state (2/2)

26.4.9 Arbitration

When several master devices simultaneously output a start condition, communication with the master devices continues until the data differs, while adjusting the clocks. An example where two masters simultaneously output a start condition and arbitration is conducted is described below.

This example assumes that one master outputs the SDAn line high (master 1) and the other master outputs the SDAn line low (master 2) while the SCLn line is low.
In this case, the communication with master 2 is prioritized, and communication is not authorized for master 1.

This kind of operation is called arbitration, and the state in which communication is not authorized is called arbitration loss. The master that lost arbitration releases the bus by setting both the SCLn and SDAn line to high impedance.

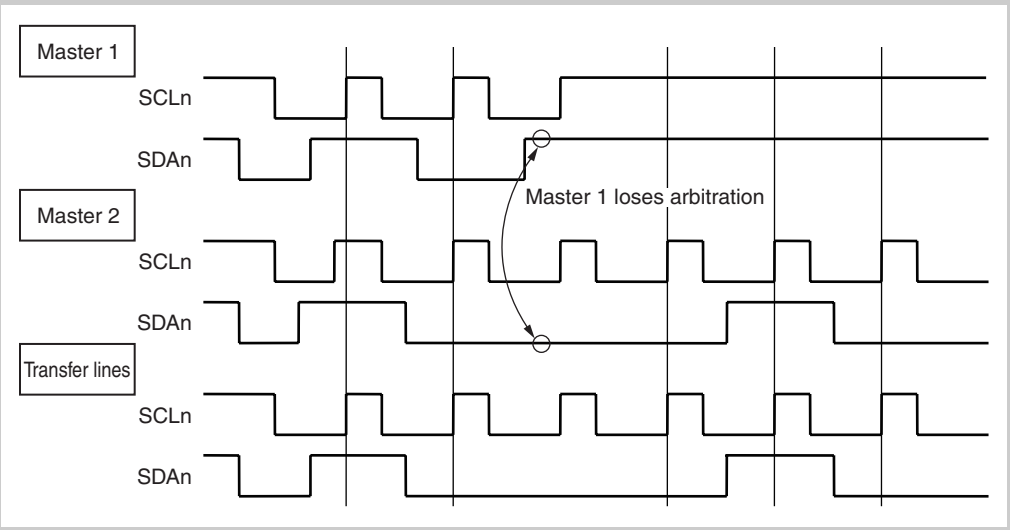


Figure 26-10 Arbitration timing example

26.5 Registers

Caution In this section, the operation when an extension code is received is omitted. For details about the extension code, see *26.6.5 "Extension code"*.

(1) IICBnDAT – IICBn data register

This register is used to transmit and receive transfer data.

Access This register can be read or written in 8-bit units.

Address <IICBn_base> + 0000_H

Initial Value 00_H. This register is initialized by any reset. This register is also initialized by changing the value of the IICBnCTL0.IICBnIICE bit from 1 to 0 or from 0 to 1.

Cautions

1. When the IICBn becomes a master in the single transfer mode or continuous transfer mode, after the IICBnTRG.IICBnSTT bit has been set to 1, writing to the IICBnDAT register is allowed only once to transfer the address and communication direction.
2. When transferring data in the single transfer mode, writing to the IICBnDAT register in communication state other than the wait state is prohibited.
3. When transferring data in the continuous transfer mode, writing to the IICBnDAT register in response to an INTIICBTIAn interrupt request signal is only allowed once.
4. When executing transmission operations in the continuous transfer mode, do not read the IICBnDAT register.
   Similarly, when performing reception operations in the continuous transfer mode, do not write to the IICBnDAT register.

7	6	5	4	3	2	1	0
IICBnDAT7	IICBnDAT6	IICBnDAT5	IICBnDAT4	IICBnDAT3	IICBnDAT2	IICBnDAT1	IICBnDAT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 26-8 IICBnDAT register contents

Bit position	Bit name	Function
7 to 0	IICBnDAT[7:0]	During reception, these bits hold the received data. During transmission, these bits write the transmit data. The prescribed procedure must be followed during access (read, write) to the IICBnDAT register. For the setting sequence, see *26.9 "Setting Sequence"*. The IICBn exits the wait state by performing access to the IICBnDAT register. • In single transfer mode - When write access to the IICBnDAT register is performed • In continuous transfer mode - When write access to the IICBnDAT register is performed - When read access to the IICBnDAT register is performed during a wait state for data transfer that is not triggered by NACK signal reception

(2) IICBnSVA – IICBn slave address register

This register stores the slave address of the IICBn bus.

Access This register can be read or written in 8-bit units.

Address <IICBn_base> + 0004_H

Initial Value 00_H. This register is initialized by any reset.

Caution Write access to the IICBnSVA register is prohibited when the value of the IICBnCTL0.IICBnIICE bit is 1.

7	6	5	4	3	2	1	0
IICBnSVA7	IICBnSVA6	IICBnSVA5	IICBnSVA4	IICBnSVA3	IICBnSVA2	IICBnSVA1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R

Table 26-9 IICBnSVA register contents

Bit position	Bit name	Function
7 to 1	IICBnSVA[7:1]	Store the slave address of the IICBn bus. Address match/address mismatch is judged by comparing the received address and the IICBnSVA register. If the received address matches the IICBnSVA register, the IICBnSTR0.IICBnSSCO bit is set to 1.

(3) IICBnCTL0 – IICBn control register 0

This register is used to control the operations of the IICBn.

Access This register can be read or written in 8-bit or 1-bit units.

Address <IICBn_base> + 0008_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
IICBn IICE	0	0	IICBn MDTX1	IICBn MDTX0	IICBn SLSI	IICBn SLWT	IICBn SLAC
R/W	R	R	R/W	R/W	R/W	R/W	R/W

Table 26-10 IICBnCTL0 register contents (1/3)

Bit position	Bit name	Function
7	IICBnIICE	Enables/disables operation of the IICBn. 0: Enables operation of IICBn. 1: Disables operation of IICBn. Synchronous reset of the following registers is executed when the value of the IICBnCTL0.IICBnIICE bit changes from 1 to 0, or the value of the IICBnCTL0.IICBnIICE bit changes from 0 to 1. • IICBnDAT and IICBnSTR0 registers When IICBnCTL0.IICBnIICE is 0, the SCLn and SDAn pins go into the high impedance state.
4	IICBnMDTX1	Specifies the transfer mode upon detection of expansion code in the slave. 0: Single transfer mode 1: Continuous transfer mode • Single transfer mode The IICBn enters a wait state after each transfer according to the setting of the IICBnCTL0.IICBnSLWT bit. • Continuous transfer mode The IICBn performs continuous communication without entering a wait state when the IICBnDAT register is read or written upon the output of the data transmit/receive interrupt request signal (INTIICBTIAn). For the operation in each mode, see *26.6 "Operation"*. **Caution** Rewrite is allowed only when IICBnCTL0.IICBnIICE is 0.
3	IICBnMDTX0	Specifies the transfer mode when the address matches between the master and slave. 0: Single transfer mode 1: Continuous transfer mode • Single transfer mode The IICBn enters a wait state after each transfer according to the setting of the IICBnCTL0.IICBnSLWT bit. • Continuous transfer mode The IICBn performs continuous communication without entering a wait state when the IICBnDAT register is read or written upon the output of the data transmit/receive interrupt request signal (INTIICBTIAn). For the operation in each mode, see *26.6 "Operation"*. **Caution** Rewrite is allowed only when IICBnCTL0.IICBnIICE is 0.

Table 26-10 IICBnCTL0 register contents (2/3)

Bit position	Bit name	Function
2	IICBnSLSI	Enables/disables status interrupt request signal (INTIICBTISn) output when a stop condition is detected. 0: Disables INTIICBTISn signal output when stop condition is detected. 1: Enables INTIICBTISn signal output when stop condition is detected. Set this bit to 1 when performing the following types of communication. - When the IICBn performs communication as a master while the communication reserve function is enabled - When the IICBn participates in communications as a slave - When the IICBn may lose in arbitration (when making the IICBn operate as a master in a multi-master environment))
1	IICBnSLWT	Controls a wait and interrupt request output timing. 0: The IICBn enters the wait state and an interrupt request is output at the falling edge of the 8th clock during single transfer. 1: The IICBn enters the wait state and an interrupt request is output at the falling edge of the 9th clock during single transfer. The IICBnCTL0.IICBnSLWT bit controls wait state transition and interrupt request output at the following timing. - 8th and 9th clocks during data transfer For the conditions for transition to the wait state, see *26.6.4 "Entering and exiting wait state"*. During address transfer, the conditions for transiting to the wait state and for interrupt request output are as follows, regardless of the setting of the IICBnCTL0.IICBnSLWT bit. • In single transfer mode - Master: A data transmit/receive interrupt request signal (INTIICBTIAn) is output and the IICBn enters the wait state upon detection of the falling edge of the 9th clock. - Slave: During an address match, the INTIICBTIAn signal is output and the IICBn enters the wait state upon detection of the falling edge of the 9th clock. During address mis-match, the INTIICBTIAn signal is not output and the IICBn does not enter the wait state. • In continuous transfer mode In the continuous transfer mode, transition to the wait state is not affected by the setting of the IICBnCTL0.IICBnSLWT bit. - Reception: The IICBn enters the wait state at the falling edge of the 8th clock. - Transmission: The IICBn enters the wait state at the falling edge of the 9th clock. **Caution** During single transfer mode, rewriting this bit is allowed only when IICBnCTL0.IICBnIICE is 0 or while in a wait period.

Table 26-10 IICBnCTL0 register contents (3/3)

Bit position	Bit name	Function
0	IICBnSLAC	Controls acknowledge signal output. 0: Disables acknowledge signal output. Master:The acknowledge signal is not output during data reception (SDAn = “H”). Slave:The acknowledge signal is not output during data transfer when an address match occurs (SDAn = “H”). 1: Enables acknowledge signal output. Master:The acknowledge signal is output during data reception (SDAn = “L”). Slave:The acknowledge signal is output during data transfer when an address match occurs (SDAn =“L”). When the IICBn is operating as a slave, in the case of an address match, no acknowledge signal is output during address transfer regardless of the value of the IICBnCTL0.IICBnSLAC bit (SDAn = “L”). also, no acknowledge signal is output (SDAn = “H”) while the IICBn is transmitting data or when it does not participate in communications.

(4) IICBnCTL1 – IICBn control register 1

This register controls the operation of the IICBn.

Access This register can be read or written in 8-bit units.

Address <IICBn_base> + 0020_H

Initial Value 00_H. This register is initialized by any reset.

Caution Write access to the IICBnCTL1 register is prohibited when the value of the IICBnCTL0.IICBnIICE is 1.

7	6	5	4	3	2	1	0
IICBn MDSC	IICBn LGDF2	IICBn LGDF1	IICBn LGDF0	IICBn MDLB	0	IICBn SLSE	IICBn SLRS
R/W	R/W	R/W	R/W	R/W	R	R/W	R/W

Table 26-11 IICBnCTL1 register contents (1/2)

Bit position	Bit name	Function
7	IICBnMDSC	Specifies the operation mode for the IICBn. 0: Standard mode (SCL clock frequency: 100 kHz max.) 1: Fast mode (SCL clock frequency: 400 kHz max.)
6 to 4	IICBnLGDF[2:0]	Specify the digital filter sampling frequency. Note that the digital filter can be used only in the fast mode. 000:Does not use digital filter. SCLn and SDAn are used without passing through the digital filter in the IICBn. The digital filter circuit operations are stopped. Other than above:Uses digital filter. SCLn and SDAn are used passing through the digital filter in the IICBn. When using a digital filter, set bits IICBnCTL1.IICBnLGDF[2:0] as follows.

IICBnCTL1.IICBnLGDF[2:0] bits	Frequency
001	Minimum frequency [a] ≤ PCLK ≤ 20 MHz
010	20 MHz < PCLK ≤ 40 MHz
011	40 MHz < PCLK ≤ 60 MHz
100	60 MHz < PCLK ≤ 80 MHz
101	80 MHz < PCLK ≤ 100 MHz
110, 111	Setting prohibited

a) A list of the minimum frequencies by setting is shown below.

Operation mode (IICBnCTL1.IICBnMDSC)	When digital filter used (IICBnCTL1.IICBnLGDF[2:0] bits = 000)	When digital filter used (IICBnCTL1.IICBnLGDF[2:0] bits ≠ 000)
Standard mode (0)	1.0 MHz	Use prohibited
Fast mode (1)	3.5 MHz	4.0 MHz

Table 26-11 IICBnCTL1 register contents (2/2)

Bit position	Bit name	Function
3	IICBnMDLB	Specifies the loop back mode. 0: Do not loop back. 1: Loop back. By setting the IICBnCTL1.IICBnMDLB bit, the output serial clock signal (SCLn) and serial transmit/receive data signal (SDAn) are looped back and used as the input serial clock signal (SCLn) and input serial transmit/receive data signal (SDAn). The output SCLn and SDAn immediately before output will be looped back. Note that both SCLn and SDAn are high level if the IICBnCTL1.IICBnMDLB bit is “1”.
1	IICBnSLSE	Enables/disables start condition output in the initial communication state. 0: Disables start condition output in the initial communication state. 1: Enables start condition output in the initial communication state. If the IICBnCTL1.IICBnSLSE bit is set to 1, a start condition can be output by setting the IICBnTRG.IICBnSTT bit to 1 in the initial communication state (from when the IICBnCTL0.IICBnIICE bit is set to 1 until detection of a stop condition). The IICBnCTL1.IICBnSLSE bit is automatically cleared to 0 upon detection of a start condition (even without a 0 write operation). **Caution** Clear the IICBnCTL1.IICBnSLSE bit to 0 when participating in communications after other communications have started. When other communications are being performed, if the IICBnTRG.IICBnSTT bit has been set to 1 with the IICBnCTL1.IICBnSLSE bit set to 1, the other communications may be damaged.
0	IICBnSLRS	Enables/disables the communication reserve function. 0: Enables communication reserve function. 1: Disables communication reserve function. Communication reserve function enabled state: If the IICBnCTL1.IICBnSLRS bit is cleared to 0 while the IICBn is not operating as a master, the communication reserve state can be set by setting the IICBnTRG.IICBnSTT bit to 1 while the bus is being used. Whether the communication reserve state is set can be confirmed by checking the IICBnSTR0.IICBnSSRS bit. Communication reserve function disabled state: If the IICBnTRG.IICBnSTT bit is set to 1 while the IICBn is not participating in communications as a master and the bus is being used, the value of the IICBnSTR0.IICBnSTCF becomes 1 and communication reservation is not done.

(5) IICBnWL – IICBn low level width setting register

This register is used to set the low level width of the serial clock register (SCLn).

Access This register can be read or written in 16-bit units.

Address <IICBn_base> + 0024_H

Initial Value $03FF_H$. This register is initialized by any reset.

Caution Write access to the IICBnWL register is prohibited when the value of the IICBnCTL0.IICBnIICE bit is 1.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	IICBnWL9	IICBnWL8
R	R	R	R	R	R	R/W	R/W

7	6	5	4	3	2	1	0
IICBnWL7	IICBnWL6	IICBnWL5	IICBnWL4	IICBnWL3	IICBnWL2	IICBnWL1	IICBnWL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 26-12 IICBnWL register contents

Bit position	Bit name	Function
9 to 0	IICBnWL[9:0]	Specify the t_{LOW} period (low level width of the SCLn clock) of the I^2C bus specification. The value of the IICBnWL register is used to determine the serial output timing of other I^2C bus specifications. For the serial output timing setting conditions, see *Table 26-14 "Conditions for generating serial output timing" on page 1983*.

(6) IICBnWH – IICBn high-level width setting register

This register is used to set the high level width of the serial clock signal (SCLn).

Access This register can be read or written in 16-bit units.

Address <IICBn_base> + 0028_H

Initial Value $03FF_H$. This register is initialized by any reset.

Caution Write access to the IICBnWH register is prohibited when the value of the IICBnCTL0.IICBnIICE bit is 1.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	IICBnWH9	IICBnWH8
R	R	R	R	R	R	R/W	R/W

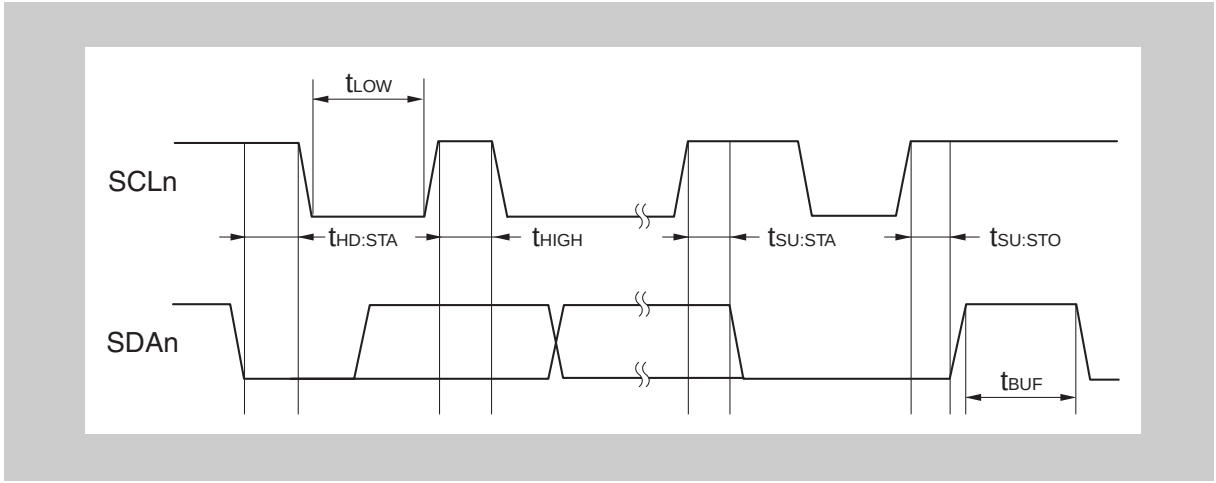
7	6	5	4	3	2	1	0
IICBnWH7	IICBnWH6	IICBnWH5	IICBnWH4	IICBnWH3	IICBnWH2	IICBnWH1	IICBnWH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 26-13 IICBnWH register contents

Bit position	Bit name	Function
9 to 0	IICBnWH[9:0]	Specify the t_{HIGH} period (high level width of the SCLn clock) of the I^2C bus specification. The value of the IICBnWH register is used to determine the serial output timing of other I^2C bus specifications. For the serial output timing setting conditions, see *Table 26-14 "Conditions for generating serial output timing" on page 1983*.

Table 26-14 Conditions for generating serial output timing

Symbol	Description	Standard mode	Fast mode
$t_{HD:STA}$	Start condition hold time	$\frac{IICB0WH}{PCLK}$	$\frac{IICB0WH}{PCLK}$
t_{LOW}	SCL low-level width period	$\frac{IICB0WL}{PCLK}$	$\frac{IICB0WL}{PCLK}$
t_{HIGH}	SCL high-level width period	$\frac{IICB0WH}{PCLK}$	$\frac{IICB0WH}{PCLK}$
$t_{SU:STA}$	Start condition setup time	$\frac{IICB0WL}{PCLK}$	$\frac{IICB0WL}{PCLK}$
$t_{SU:STO}$	Stop condition setup time	$\frac{IICB0WH}{PCLK}$	$\frac{IICB0WH}{PCLK}$
t_{BUF}	Bus free time (interval between stop condition and start condition)	$\frac{IICB0WL}{PCLK}$	$\frac{IICB0WL}{PCLK}$
$t_{HD:DAT}$	Data hold time	$\frac{IICB0WL[9:2]}{PCLK}$	$\frac{IICB0WL[9:2]}{PCLK}$



(a) Setting transfer clock by using IICBnWL and IICBnWH registers

The various timings in compliance with the IICB bus specifications can be set by setting the IICBnWL register and IICBnWH register.

- Setting transfer clock on master side

$$\text{Transfer clock (Hz)} = \frac{\text{PCLK}}{(\text{IICBnWL} + \text{IICBnWH} + \text{PCLK}\ (t_R + t_F)}$$

At this time, the optimal setting values of IICBnWL and IICBnWH are as follows.
(The fractional parts of all setting values are rounded up.)

– In the fast mode

$$\text{IICBnWL} = \frac{0.52}{\text{Transfer clock}} \times \text{PCLK}$$

$$\text{IICBnWH} = \left(\frac{0.48}{\text{Transfer clock}} - t_R - t_F\right) \times \text{PCLK}$$

– In the standard mode

$$\text{IICBnWL} = \frac{0.47}{\text{Transfer clock}} \times \text{PCLK}$$

$$\text{IICBnWH} = \left(\frac{0.53}{\text{Transfer clock}} - t_R - t_F\right) \times \text{PCLK}$$

Caution The data hold time must be 0.9 μs or less in the fast mode and 3.45 μs or less in the standard mode.

Note The data hold time is determined by the IICBWL register setting as follows:

Data hold time = IICBnWL.IICBnWL[9:2] / PCLK

- Setting IICBnWL and IICBnWH on slave side
  (The fractional parts of all setting values are rounded up.)
  - In the fast mode

    IICBnWL = 1.3 μs × PCLK

    IICBnWH = (1.2 μs – t_R – t_F) × PCLK
  - In the standard mode

    IICBnWL = 4.7 μs × PCLK

    IICBnWH = (5.3 μs – t_R – t_F) × PCLK

Note IICBnWL: IICBn low-level width setting register
IICBnWH: IICBn high-level width setting register
t_F: SDAn and SCLn signal falling times
t_R: SDAn and SCLn signal rising times
PCLK: Frequency of the clock supplied to the IICBn
f_{CLK}: SCL clock frequency

(7) IICBnTRG – IICBn trigger register

This register is used to set the IICBn trigger.

Access This register can be read or written in 8-bit or 1-bit units.

Address <IICBn_base> + $000C_H$

Initial Value 0000_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	IICBn LRET	IICBn WRET	IICBn STT	IICBn SPT
R	R	R	R	R/W	R/W	R/W	R/W

Table 26-15 IICBnTRG register contents (1/4)

Bit position	Bit name	Function
3	IICBnLRET	Communication exit trigger bit 0: The read value is always 0, and writing 0 is ignored. 1: The IICBn exits the current communication and enters the wait state. This bit is automatically cleared to 0 following execution. The following occurs when IICBnTRG.IICBnLRET is 1. - SCLn and SDAn each go into high impedance (communication wait state). - Bits IICBnSSMS, IICBnSSDR, IICBnSSWT, IICBnSSEX, IICBnSSC0, IICBnSSTR, IICBnSSAC, IICBnSSRS, and IICBnSSST of the IICBnSTR0 register are cleared to 0. - When IICBnTRG.IICBnSTT = 1 (start condition output preparation) or IICBnTRG.IICBnSPT =1 (stop condition output preparation) has been set, output of a start condition or stop condition is stopped. The communication reserved state is released if the IICBn exits the communication in the communication reserved state. If it is necessary for the IICBn to operate a master again after this, the IICBnTRG.IICBnSTT bit must be set to 1 again. **Caution** If IICBnTRG.IICBnLRET is set to 1 during master operation (IICBnSTR0.IICBnSSMS = 1), the bus is released. Because serial clock output stops, problems occur during communication on the slave side.
2	IICBnWRET	This is the trigger bit for exiting the wait state. 0: Does not exit the wait state. 1: Exits the wait state and resumes communication. This bit is automatically cleared following execution. If the IICBn have exited the wait state by setting the IICBnTRG.IICBnWRET bit to 1 during the wait state triggered by the falling edge of the 9th clock, the IICBnSTR0.IICBnSSTR bit is cleared to 0 and SDAn goes into high impedance (this enables the external master to output a stop condition or start condition.) If the IICBn is not in the wait state (IICBnSTR0.IICBnSSWT = 0), setting this bit to 1 has no meaning. There are other conditions for exiting the wait state in addition to the setting of this bit. For details, see *26.6.4 “Entering and exiting wait state”*.

Table 26-15 IICBnTRG register contents (2/4)

Bit position	Bit name	Function
1	IICBnSTT	Start condition trigger bit 0: Does not output a start condition. 1: Outputs a start condition (This bit is automatically cleared to 0 after it has been set to 1. The IICBnTRG.IICBnSTT bit can be set to 1 under the following conditions: [1] IICBnSTR0.IICBnSSMS bit = Master state (1) • Single transfer mode - During wait state triggered by the falling edge of the 9th clock (both address transfer and data transfer) - During data reception, only after clearing the IICBnCTL0.IICBnSLAC bit to 0 to report the end of reception to the slave • Continuous transfer mode - During wait state triggered by the falling edge of the 9th clock of address transfer - During data transfer - During data reception, only after clearing the IICBnCTL0.IICBnSLAC bit to 0 to report the end of reception to the slave In the case of the wait period during the 9th clock, following wait cancellation, and in all other cases, upon detecting the falling edge of the 9th clock, SDAn and SCLn are set to the high level after the low-level width period of the SCLn clock, and then, when SDAn is set to the low level after waiting for the start condition setup time to elapse, a start condition is output. Next, SCLn is set to the low level after the start condition hold time has elapsed. For each time, see *Table 26-14 "Conditions for generating serial output timing" on page 1983*. [2] Slave state or communication wait state (IICBnSTR0.IICBnSSMS = 0) • IICBnSTR0.IICBnSSBS bit = 0 (bus release state) After the bus free time elapses, a start condition is output when SDAn is changed from the high level to the low level while SCLn is high level. (At this time, SCLn outputs a high level signal.) Next, SCLn is set to the low level after the start condition hold time has elapsed. For each time, see *Table 26-14 "Conditions for generating serial output timing" on page 1983*. • IICBnSTR0.IICBnSSBS bit = 1 (bus communication state) This status indicates that communication is performed on the bus while the IICBn is not operating as a master. - When communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0): A start condition is output after the bus has been released (the stop condition has been detected) and the bus free time has elapsed. However, even if the bus free time has not elapsed, upon detecting a start condition, SDAn is immediately set to the low level without waiting for the bus free time to elapse. For each time, see *Table 26-14 "Conditions for generating serial output timing" on page 1983*. - When communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1): The IICBnSTR0.IICBnSTCF bit is set to 1 and a start condition is not output.

Table 26-15 IICBnTRG register contents (3/4)

Bit position	Bit name	Function
1	IICBnSTT	**Caution** [2] shows the operations according to the value of the IICBnSTR0.IICBnSSBS bit when the IICBnTRG.IICBnSTT bit is 0. Even if the IICBnTRG.IICBnSTT bit is set to 1 after checking the value of the IICBnSTR0.IICBnSSBS bit through register read, the value of IICBnSTR0.IICBnSSBS may differ from its value when it was checked. The output processing of the start condition is started by setting the IICBnTRG.IICBnSTT bit to 1, but upon detection of the following states, output processing of the start condition is stopped and the start condition is not output. - When 0 is written to the IICBnCTL0.IICBnIICE bit - When 1 is written to the IICBnTRG.IICBnLRET bit - Upon detection of arbitration loss - When 1 is written to the IICBnTRG.IICBnSPT bit after 1 is written to the IICBnTRG.IICBnSTT bit while the IICBn is operating as a master in the continuous transfer mode - When 1 is written to the IICBnTRG.IICBnSTT and IICBnTRG.IICBnSPT bits during the same data transfer period while the IICBn is operating as a master in the continuous transfer mode (In this case, writing 1 to the IICBnTRG.IICBnSTT bit is enabled.) **Cautions** 1. When start in the initial communication state is enabled (IICBnCTL1.IICBnSLSE bit = 1), the start condition is output regardless of the bus status when the IICBnTRG.IICBnSTT bit is set to 1. If other communications are performed at that time, they may be damaged. 2. Setting the IICBnTRG.IICBnSTT bit at the same time as the IICBnTRG.IICBnSPT bit is prohibited.
0	IICBnSPT	Stop condition trigger bit 0: Does not output a stop condition. 1: Outputs a stop condition (This bit is automatically cleared after it has been set to 1). The IICBnTRG.IICBnSPT bit can be set to 1 under the following conditions while the IICBn is performing communication as a master. • Single transfer mode - Wait state triggered by the falling edge of the 9th clock (both address transfer and data transfer) - During data reception, only after clearing the IICBnCTL0.IICBnSLAC bit to 0 to report the end of reception to the slave • Continuous transfer mode The IICBnTRG.IICBnSPT bit can be set to 1 in the following states. - During the wait state triggered by the falling edge of the 9th clock of address transfer - During data transfer - Detection of a NACK signal (IICBnSTR0.IICBnSSAC bit = 0) during the wait state triggered by the falling edge of the 9th clock for during data reception A stop condition can be output with the following procedure. (If the IICBn is in the wait state, after exiting the wait state) SCLn is released when SDAn has output a low level, and SCLn = high level, SDAn is low level are waited for. Then, following the lapse of the $t_{SU:STO}$ time, a stop condition is output by setting SDAn to high level.

Table 26-15 IICBnTRG register contents (4/4)

Bit position	Bit name	Function
0	IICBnSPT	The output processing of the stop condition is started by setting the IICBnTRG.IICBnSPT bit to 1, but upon detection of the following states, output processing of the stop condition is stopped and the stop condition is not output. - When 0 is written to the IICBnCTL0.IICBnIICE bit - When 1 is written to the IICBnTRG.IICBnLRET bit - Upon detection of a stop condition - Upon detection of arbitration loss - When 1 is written to the IICBnTRG.IICBnSTT bit after IICBnTRG.IICBnSPT has been set to 1 while the IICBn is operating as a master in the continuous transfer mode **Cautions** 1. Setting the IICBnTRG.IICBnSPT bit to 1 is prohibited during slave operation (IICBnSTR0.IICBnSSMS bit = 0) 2. Setting the IICBnTRG.IICBnSPT bit to 1 at the same time as the IICBnTRG.IICBnSTT bit is prohibited.

(8) IICBnSTR0 – IICBn status register 0

This register indicates the statuses of the IICBn and the bus.

Access This register can be read only in 16-bit units. However, when IICBnCTL0.IICBnIICE is 0, this register can also be write accessed.

Address <IICBn_base> + 0010_H

Initial Value 0000_H. This register is initialized by any reset. This register is also initialized by changing the value of the IICBnCTL0.IICBnIICE bit from 1 to 0 or from 0 to 1.

15	14	13	12	11	10	9	8
IICBn SSMS	0	IICBn SSDR	IICBn SSWT	IICBn SSEX	IICBn SSCO	IICBn SSTR	IICBn SSAC
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
IICBn SSRS	IICBn SSBS	IICBn SSST	IICBn SSSP	0	0	IICBn STCF	IICBn ALDF
R	R	R	R	R	R	R	R

Table 26-16 IICBnSTR0 register contents (1/6)

Bit position	Bit name	Function
15	IICBnSSMS	Master state check flag: Indicates that the IICBn is operating as a master. 1: Indicates that the IICBn is operating as a master. Setting condition: Upon detection of a start condition after 1 is written to the IICBnTRG.IICBnSTT bit Clearing conditions: • When 1 is written to the IICBnTRG.IICBnLRET bit • Upon detection of a stop condition • Upon detection of arbitration loss If a setting condition coincides with a clearing condition, the clearing condition takes priority.
13	IICBnSSDR	IICBnDAT register status flag 1: Indicates that data in the IICBnDAT register remains unprocessed. During reception operation:Received data remains unread in the IICBnDAT register. During transmission operation:Data written to the IICBnDAT register has not been transferred to the shift register. **Note:** This bit is not set while in the single transfer or continuous transfer mode. This is because the IICBnDAT register cannot be written during the wait period while in the single transfer mode (transmission). Setting condition: • When the IICBnDAT register is written during address transfer and data transfer while the IICBnSTR0.IICBnSSWT bit is 0 (Note that, even if the IICBnSTR0.IICBnSSWT bit is 0, the IICBnSSDR bit is not set to 1 if address data is written to the IICBnDAT register while the IICBn is operating as a master, because the address data is directly transferred to the shift register in this case.) • At the falling edge of the 9th clock after an address match with a slave • While IICBnCTL0.IICBnSLWT = 0 and single mode reception is being performed, at the falling edge of the 8th clock during data reception

Table 26-16 IICBnSTR0 register contents (2/6)

Bit position	Bit name	Function
13	IICBnSSDR	• At the falling edge of the 8th clock while in the continuous transfer mode (reception), regardless of the IICBnCTL0.IICBnSLWT bit value • While IICBnCTL0.IICBnSLWT = 1, at the falling edge of the 9th clock during data reception Clearing conditions: • Clearing conditions given priority over setting conditions - When 1 is written to the IICBnTRG.IICBnLRET bit - Upon detection of arbitration loss - At the falling edge of the 9th clock during address transfer while the IICBn is operating as a master - At the falling edge of the 8th clock during data transmission while IICBnCTL0.IICBnSLWT = 0 and continuous transmission is being performed - At the falling edge of the 9th clock during data transmission while IICBnCTL0.IICBnSLWT = 1 and continuous transmission is being performed • Clearing condition for which setting conditions are given priority (while in the continuous transfer mode (transmission) - When the IICBnDAT register is read while the shift register does not have any received data that must be transferred to the IICBnDAT register
12	IICBnSSWT	Wait state flag 1: Indicates that the IICBn is in the wait state. Setting condition: ■ In single transfer mode <Common to master/slave> • During data transfer, upon detection of the falling edge of the 8th clock with IICBnCTL0.IICBnSLWT = 0 • During data transfer, upon detection of the falling edge of the 9th clock with IICBnCTL0.IICBnSLWT = 1 <Master> • When the IICBn becomes a master (IICBnSTR0.IICBnSSMS = 1) after 1 is written to the IICBnTRG.IICBnSTT bit, and the falling edge of the first SCLn is detected without the IICBnDAT register being written • Upon detection of the falling edge of the 9th clock during address transfer <Slave> • Upon detection of the falling edge of the 9th clock during address transfer when an address match occurred ■ In continuous transfer mode <During data transfer period, common to master/slave> • During data transmission, when the data to be transmitted next has not been written - When IICBnCTL0.IICBnSLWT = 0, at the falling edge of the 8th clock during data transmission with IICBnSTR0.IICBnSSDR = 0 - When IICBnCTL0.IICBnSLWT = 1, at the falling edge of the 9th clock during data transmission with IICBnSTR0.IICBnSSDR = 0 • During data reception, when the previous received data has not been read - When IICBnCTL0.IICBnSLWT = 0, at the falling edge of the 8th clock during data reception with IICBnSTR0.IICBnSSDR = 1 - When IICBnCTL0.IICBnSLWT = 1, at the falling edge of the 9th clock during data reception with IICBnSTR0.IICBnSSDR = 1 - Upon NACK detection (However, only if 1 has not been written to IICBnTRG.IICBnSTT or IICBnTRG.IICBnSPT while the IICBn is operating as a master)

Table 26-16 IICBnSTR0 register contents (3/6)

Bit position	Bit name	Function
12	IICBnSSWT	During address transfer period, operating as master> - When the IICBn becomes a master (IICBnSTR0.IICBnSSMS = 1) after 1 is written to the IICBnTRG.IICBnSTT bit, and the first falling edge of SCLn is detected without the IICBnDAT register being written - Upon NACK detection (However, only if 1 has not been written to IICBnTRG.IICBnSTT or IICBnTRG.IICBnSPT) <During address transfer period, operating as slave> - Upon detection of the falling edge of the 9th clock while IICBnSTR0.IICBnSSTR bit is 0 during address transfer when an address match occurred - Upon NACK detection Clearing conditions: • Clearing conditions given priority over setting conditions - When 1 is written to the IICBnTRG.IICBnLRET bit - When 1 is written to the IICBnTRG.IICBnSTT bit while the IICBn is operating as a master in the continuous transfer mode - When 1 is written to the IICBnTRG.IICBnSPT bit while the IICBn is operating as a master in the continuous transfer mode - When the IICBnDAT register is written while the IICBn is performing transmission in the continuous transfer mode - During the wait state triggered by the falling edge of the 8th clock, when the IICBnDAT register is read while reception is performed in the continuous transfer mode - During the wait state triggered by the falling edge of the 9th clock, when the IICBnDAT register is read while the IICBn is performing reception in the continuous transfer mode and an acknowledge signal (ACK) has been received • Clearing condition for which setting conditions are given priority - When 1 is written to the IICBnTRG.IICBnWRET bit - When 1 is written to the IICBnTRG.IICBnSTT bit while the IICBn is operating as a master in the single transfer mode - When 1 is written to the IICBnTRG.IICBnSPT bit while the IICBn is operating as a master in the single transfer mode - When the IICBnDAT register is written while the IICBn is performing reception in the single transfer mode **Caution** If the IICBn exits the wait state that was triggered by the falling edge of the 9th clock by writing 1 to the IICBnTRG.IICBnWRET bit, the IICBnSTR0.IICBnSSTR bit is cleared to 0 and the bus is released (both SCLn and SDAn go into high impedance).
11	IICBnSSEX	Expansion code reception detection flag 1: Indicates that an expansion code has been received. Setting condition: Upon detection of the falling edge of the 8th clock while transferring received address data whose higher 4 bits are either 0000 or 1111 Clearing conditions: • When 1 is written to the IICBnTRG.IICBnLRET bit • Upon detection of a stop condition • Upon detection of a start condition **Caution** When the expansion codes match, the processing after the interrupt differs according to the ensuing data, and therefore is dependent on software processing.

Table 26-16 IICBnSTR0 register contents (4/6)

Bit position	Bit name	Function
10	IICBnSSCO	Address match detection flag 1: Indicates that an address that matches the IICBnSVA register has been detected. Setting condition: Upon detection of the falling edge of the 8th clock while transferring a received address that matches the IICBnSVA register Clearing conditions: • When 1 is written to the IICBnTRG.IICBnLRET bit • Upon detection of a stop condition • Upon detection of a start condition
9	IICBnSSTR	Transmission status detection flag 1: Indicates that data is being transmitted to the serial data bus. Setting condition: <Master> - Upon detection of a start condition after 1 is written to the IICBnTRG.IICBnSTT bit <Slave> - Upon detection of the falling edge of the 8th clock following reception of 1 to R/$\overline{W}$ bit during address transfer when an address match occurred Clearing conditions: <Common to master/slave> - When 1 is written to the IICBnTRG.IICBnLRET bit - Upon detection of a stop condition - When 1 is written to the IICBnTRG.IICBnWRET bit during the wait state triggered by the falling edge of the 9th clock <Master> - Upon detection of the falling edge of the 8th clock following reception of 1 to R/$\overline{W}$ bit during address transfer - Upon detection of arbitration loss <Slave> - Upon detection of a start condition
8	IICBnSSAC	Acknowledge ($\overline{ACK}$) detection flag 1:Indicates that an acknowledge signal has been detected. Setting condition: Upon detection of the falling edge of SCLn when a low level has been received at the $\overline{ACK}$ bit during participation in communications Clearing conditions: • When 1 is written to the IICBnTRG.IICBnLRET bit • Upon detection of the rising edge of SCLn **Caution** The value of the IICBnSTR0.IICBnSSAC bit changes regardless of whether or not an interrupt has occurred.
7	IICBnSSRS	Communication reserve state flag 0: Not communication reserve state 1: Communication reserve state Setting condition: When 1 is written to the IICBnTRG.IICBnSTT bit during bus communication while the IICBn is not operating as a master, in the communication reserve function enabled state (IICBnCTL1.IICBnSLRS = 0) Clearing conditions: • When 1 is written to the IICBnTRG.IICBnLRET bit • When IICBnSTR0.IICBnSSMS = 1

Table 26-16 IICBnSTR0 register contents (5/6)

Bit position	Bit name	Function
6	IICBnSSBS	IICBn bus status flag 0: Bus released state (initial communication state when IICBnCTL1.IICBnSLSE = 1) 1: Bus communication state (initial communication state when IICBnCTL1.IICBnSLSE = 0) Setting condition: • Upon detection of a start condition • When 1 is written to the IICBnCTL0.IICBnIICE bit when IICBnCTL1.IICBnSLSE = 0 Clearing conditions: Upon detection of a stop condition **Note** The IICBnSTR0.IICBnSSBS bit operates whether or not the IICBn is participating in communications.
5	IICBnSSST	Start condition detection flag 1: Indicates that a start condition has been detected. Setting condition: Upon detection of a start condition Clearing conditions: • When 1 is written to the IICBnTRG.IICBnLRET bit • Upon detection of a stop condition • Upon detection of the rising edge of SCLn following the end of address transfer **Note** The IICBnSTR0.IICBnSSST bit operates whether or not the IICBn is participating in communications.
4	IICBnSSSP	Stop condition detection flag 1: Indicates that a stop condition has been detected. Setting condition: Upon detection of a stop condition Clearing conditions: Upon detection of the falling edge of the first SCLn following start condition detection **Note** The IICBnSTR0.IICBnSSSP bit operates whether or not the IICBn is participating in communications.
1	IICBnSTCF	IICBnTRG.IICBnSTT bit clear flag 1: Indicates that the IICBnTRG.IICBnSTT bit has been cleared because start condition output failed. Setting condition: When 1 is written to the IICBnTRG.IICBnSTT bit during bus communication when the IICBn is not operating as a master, in the communication reserve function disabled state (IICBnCTL1.IICBnSLRS = 1) **Caution** Even if the bus is released in the external bus state, this bit is set to 1 when 1 is written to the IICBnTRG.IICBnSTT bit if the communication reserve function is disabled, unless the IICBn recognizes the bus release state (IICBnSTR0.IICBnSSBS = 1). Clearing condition: When 1 is written to the IICBnSTRC.IICBnCLSF bit

Table 26-16 IICBnSTR0 register contents (6/6)

Bit position	Bit name	Function
0	IICBnALDF	Arbitration loss detection flag 1: Indicates that an arbitration loss has been detected. Setting condition: Upon detection of arbitration loss Clearing condition: When 1 is written to the IICBnSTRC.IICBnCLAF bit If a setting condition coincides with a clearing condition, the setting condition takes priority. Upon detection of arbitration loss, the IICBnSTR0.IICBnSSMS and IICBnSTR0.IICBnSSTR bits are cleared to 0. (SCLn and SDAn become high level and the bus is released.) **Caution** When the IICBnSTR0.IICBnALDF bit is set to 1 due to arbitration loss, the INTIICBTIA or INTIICBTISn interrupt request signal is output. After confirming that the IICBnSTR0.IICBnALDF bit has been set to 1 with an interrupt request signal, clear the IICBnSTR0.IICBnALDF bit with the IICBnSTRC.IICBnCLAF bit. If the value of the IICBnSTR0.IICBnALDF is not cleared and remains 1, the INTIICBTISn interrupt request signal will be output at the interrupt timing, even during unrelated communication.

(9) IICBnSTR1 – IICBn status register 1

This register indicates the status of the serial bus.

Access This register is read-only, in 8-bit units.

Address <IICBn_base> + 0014_H

Initial Value 00_H. This register is initialized by any reset.

Caution The serial clock (SCLn) and serial transmit/receive data (SDAn) are also read from an external source in the loopback mode (IICBnCTL1.IICBnMDLB = 1).

7	6	5	4	3	2	1	0
0	0	0	0	0	0	IICBn SSCL	IICBn SSDA
R	R	R	R	R	R	R	R

Table 26-17 IICBnSTR1 register contents

Bit position	Bit name	Function
1	IICBnSSCL	Indicates the level of the SCLn pin (input). 0: Low level 1: High level
0	IICBnSSDA	Indicates the level of the SDAn pin (input). 0: Low level 1: High level

(10) IICBnSTRC – IICBn status clear register

This register clears the IICBnSTCF and IICBnALDF bits of the IICBnSTR0 register.

Access This register can be read or written in 16-bit units.

Address <IICBn_base> + 0018_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	0	0	0	0	0	IICBn CLSF	IICBn CLAF
R	R	R	R	R	R	R/W	R/W

Table 26-18 IICBnSTRC register contents

Bit position	Bit name	Function
1	IICBnCLSF	Clears the IICBnSTR0.IICBnSTCF bit. 1: Clears the IICBnSTR0.IICBnSTCF bit. **Note** If the IICBnSTRC.IICBnCLSF bit is read after setting data, 0 is returned.
0	IICBnCLAF	Clears the IICBnSTR0.IICBnALDF bit. 1: Clears the IICBnSTR0.IICBnALDF bit. **Caution** If writing 1 to the IICBnSTRC.IICBnCLAF bit and the setting condition of the IICBnSTR0.IICBnALDF bit occur at the same time, the setting condition of the IICBnSTR0.IICBnALDF takes priority. **Note** If the IICBnSTRC.IICBnCLAF bit is read after data setting, 0 is returned.

26.6 Operation

The IICBn supports two transfer modes, single transfer mode and continuous transfer mode.

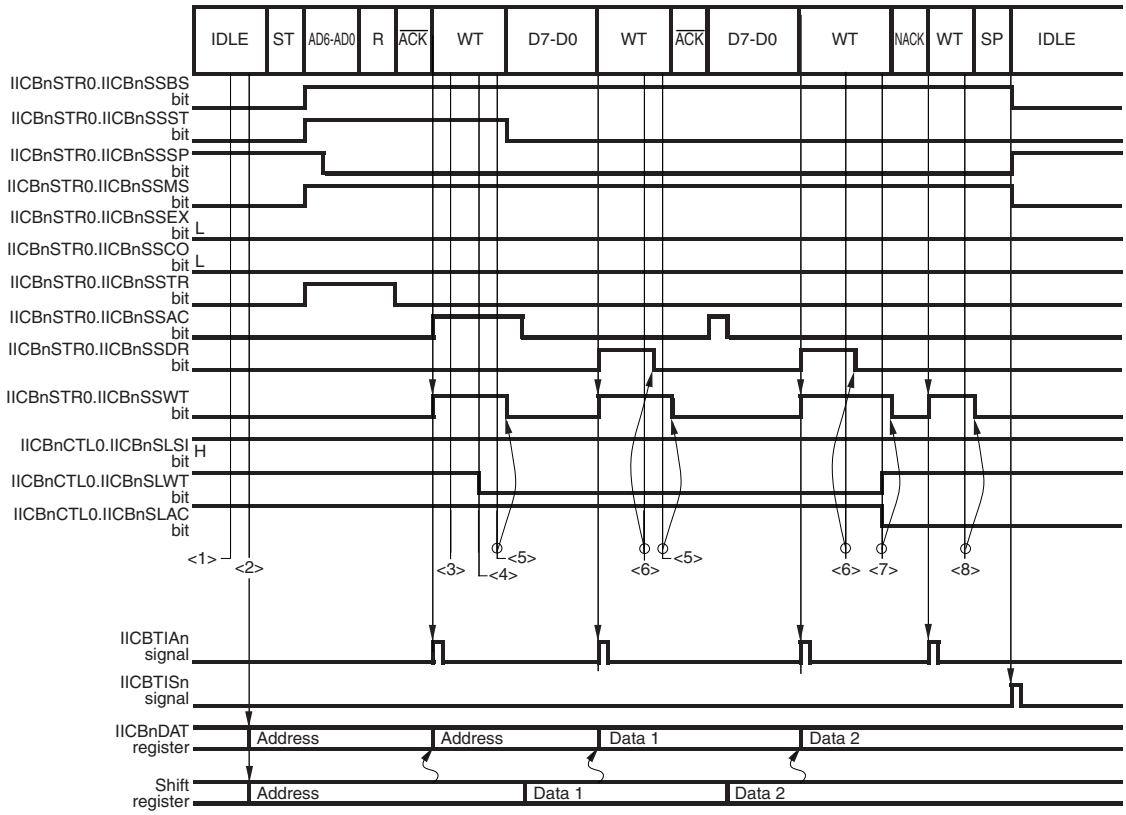
The transfer mode when the addresses of the master device and the slave device match is selected with the IICBnCTL0.IICBnMDTX0 bit, and the transfer mode when the extension code is detected by the slave device is selected with the IICBnCTL0.IICBnMDTX1 bit.

26.6.1 Single transfer mode

In the single transfer mode, a data transmit/receive interrupt request signal is output at the timing specified using the IICBnCTL0.IICBnSLW bit to make the IICBn enter the wait state, and transmit/receive data processing is performed during this wait state.

The various processing operations are described below.

(1) Example of communication in single transfer mode (master reception)



<1> Start condition output
Set the IICBnTRG.IICBnSTT bit (to 1).

<2> Address and transfer direction specification output
Set the address of the slave device and the transfer direction as 8 bits into the IICBnDAT register.

<3> Acknowledge result check
Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit using the IICBTIAn interrupt.

<4> Wait timing setting
During data reception, clear the IICBnCTL0.IICBnSLWT bit (to 0) so that the IICBn enters the wait state at the falling edge of the 8th clock.

<5> Data reception
Exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1) to start reception.

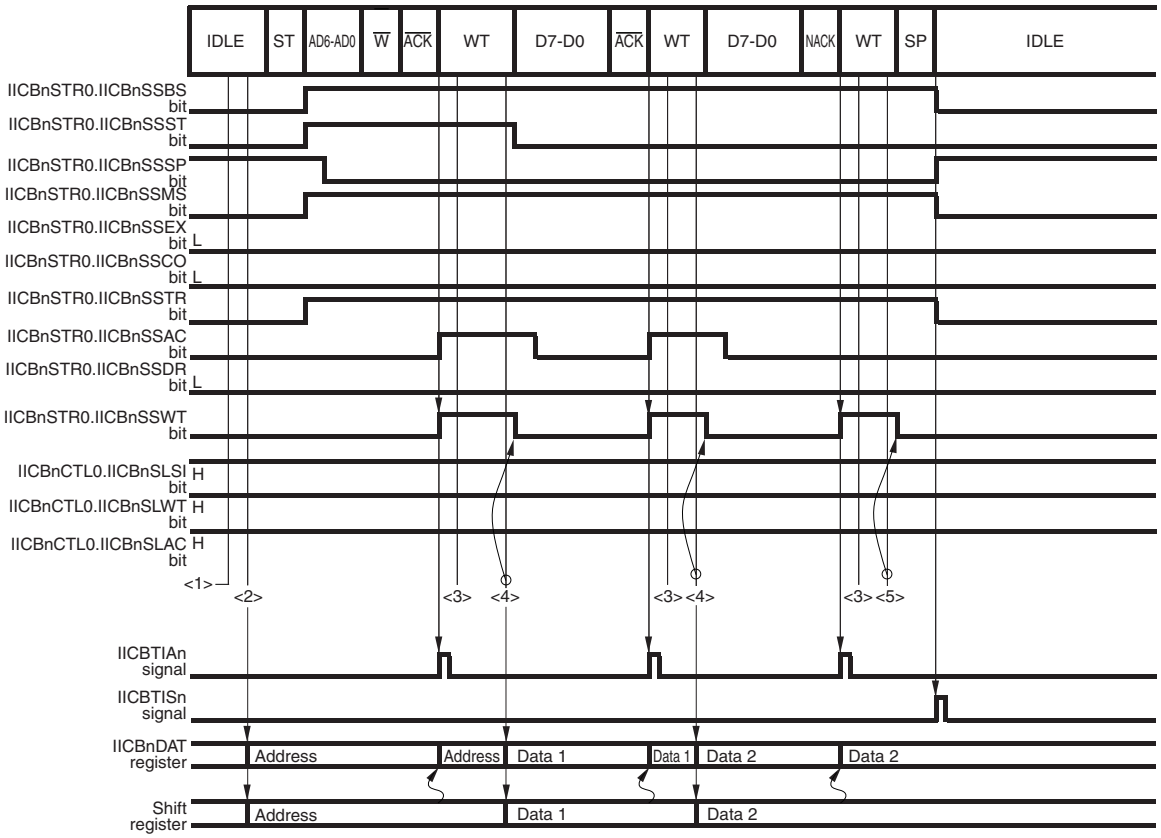
<6> Receive data load
Read the receive data from the IICBnDAT register using the IICBTIAn interrupt.

<7> Data reception completion processing

- Set the IICBnCTL0.IICBnSLWT bit to 1 and the IICBnCTL0.IICBnSLAC bit to 0.
- Next, exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1). The end of the data is notified to the transmitting device without outputting $\overline{\text{ACK}}$.

<8> Stop condition output
Set the IICBnTRG.IICBnSPT bit (to 1).

(2) Example of communication in single transfer mode (master transmission)



Note During data transmission, set the IICBnCTL0.IICBnSLWT bit (to 1) so that the IICBn enters the wait state at the falling edge of the 9th clock.

<1> Start condition output
Set the IICBnTRG.IICBnSTT bit (to 1).

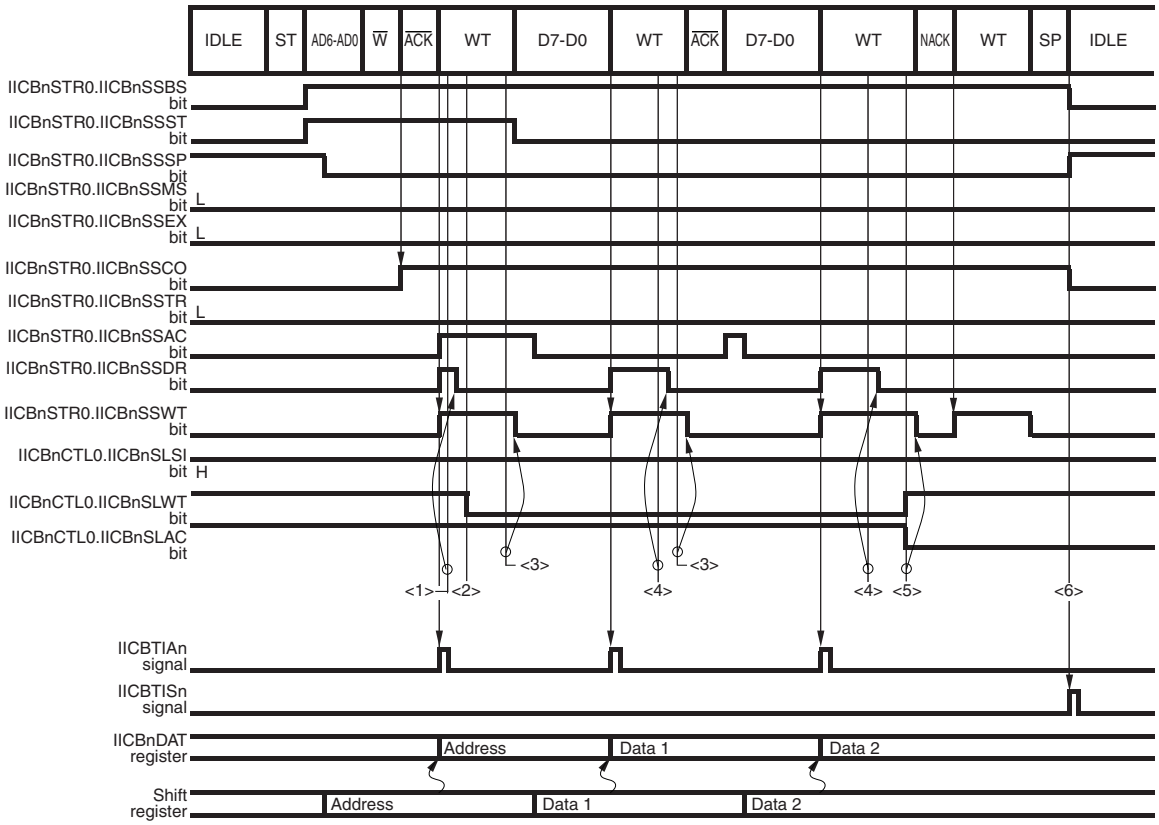
<2> Address and transfer direction specification output
Set the address of the slave device and the transfer direction as 8 bits into the IICBnDAT register.

<3> Acknowledge result check
Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit using the IICBTIAn interrupt.

<4> Data transmission
Exit the wait state by setting the transmit data into the IICBnDAT register to start transmission.

<5> Stop condition output
Set the IICBnTRG.IICBnSPT bit (to 1).

(3) Example of communication in single transfer mode (slave reception)



<1> Operation mode check in slave mode

- Check the operation mode using the IICBTIAn interrupt.
- Check the address transfer, address match, and reception operation with the IICBnSTR0.IICBnSSST, IICBnSTR0.IICBnSSCO, and IICBnSTR0.IICBnSSTR bits.
- Read the IICBnDAT register (empty read).

<2> Wait timing setting
During data reception, clear the IICBnCTL0.IICBnSLWT bit (to 0) so that the IICBn enters the wait state at the falling edge of the 8th clock.

<3> Data reception
Exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1) to start reception.

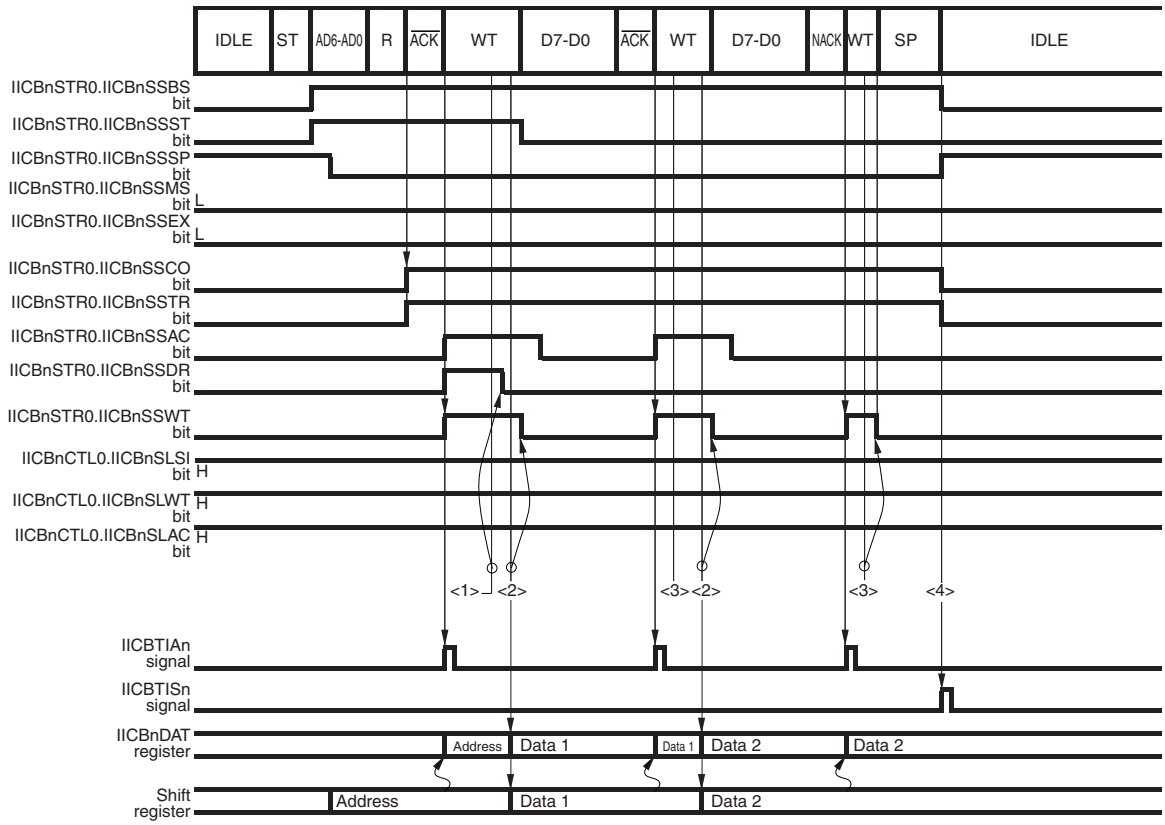
<4> Receive data load
Read the receive data from the IICBnDAT register using the IICBTIAn interrupt.

<5> Data reception completion processing

- Set the IICBnCTL0.IICBnSLWT bit to 1 and the IICBnCTL0.IICBnSLAC bit to 0.
- Next, exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1). The end of the data is notified to the transmitting device without outputting $\overline{\text{ACK}}$.

<6> Stop condition detection
Detect the stop condition using the IICBTISn interrupt.

(4) Example of communication in single transfer mode (slave transmission)



Note During data transmission, set the IICBnCTL0.IICBnSLWT bit (to 1) so that the IICBn enters the wait state at the falling edge of the 9th clock.

<1> Operation mode check in slave mode

- Check the operation mode using the IICBTIAn interrupt.
- Check the address transfer, address match, and reception operation with the IICBnSTR0.IICBnSSST, IICBnSTR0.IICBnSSCO, and IICBnSTR0.IICBnSSTR bits.
- Read the IICBnDAT register (empty read).

<2> Data transmission
Exit the wait state by setting the transmit data into the IICBnDAT register to start transmission.

<3> Acknowledge result check
Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit using the IICBTIAn interrupt.
If $\overline{\text{ACK}}$ is not output, the transmission is judged to have been completed, and the IICBn exits the wait state by setting the IICBnTRG.IICBnWRET bit (to 1).

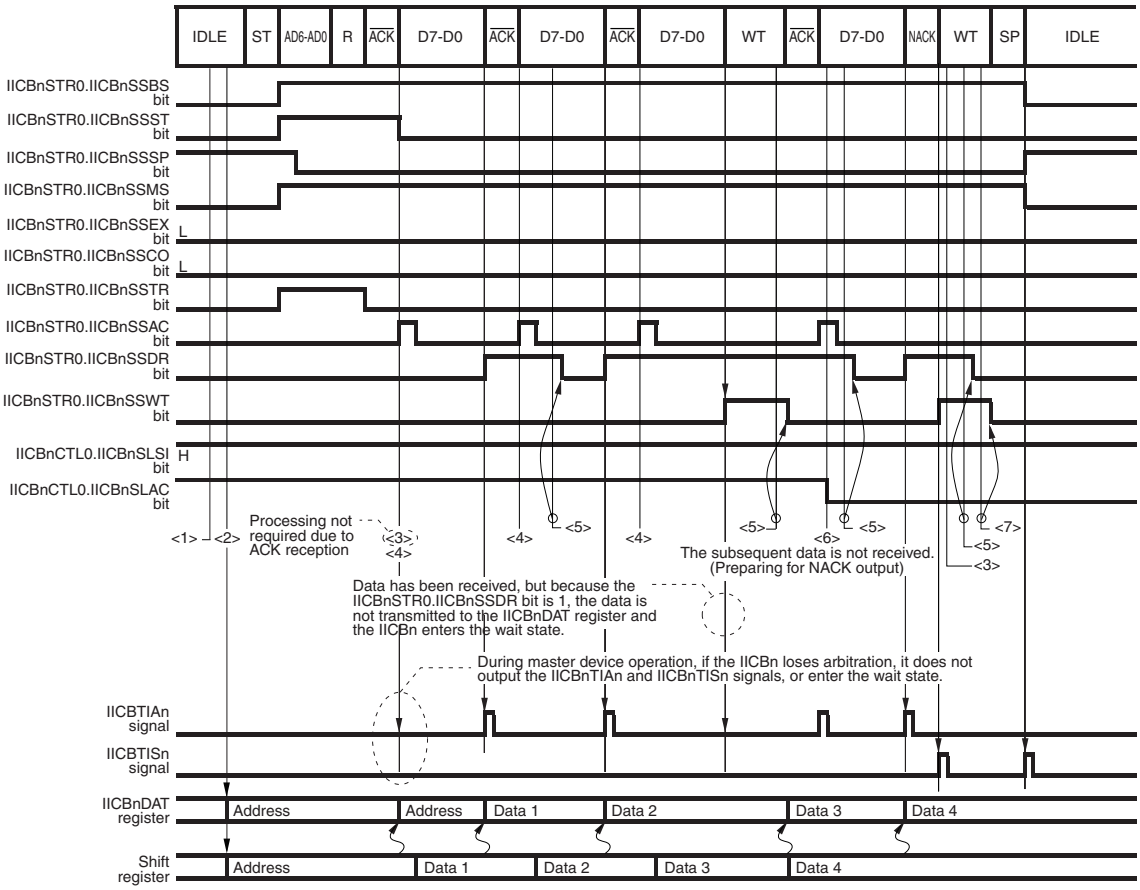
<4> Stop condition detection
Detect the stop condition using the IICBTISn interrupt.

26.6.2 Continuous transfer mode

The continuous transfer mode allows continuous communication without entering the wait state by reading/writing data to/from the IICBnDAT register each time the data transmit/receive interrupt request signal (IICBTIAn) is output.

The processing operations are described below.

(1) Example of communication in continuous transfer mode (master reception)



<1> Start condition output
Set the IICBnTRG.IICBnSTT bit (to 1).

<2> Address and transfer direction specification output
Set the address of the slave device and the transfer direction as 8 bits into the IICBnDAT register.

<3> Acknowledge result check
The IICBTISn interrupt occurs only if the slave device does not return $\overline{\text{ACK}}$. Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit.

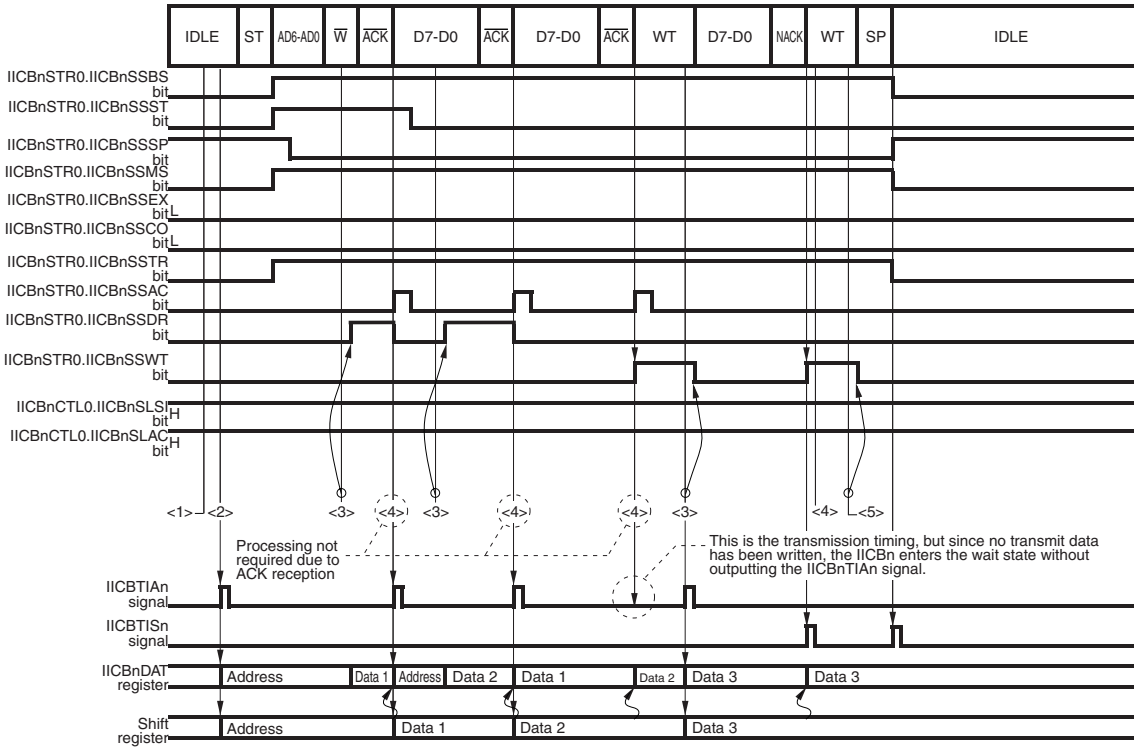
<4> Acknowledge result check
If there is no unread data in the IICBnDAT register by the time reception starts, the IICBn starts reception without entering the wait state.

<5> Receive data load
Read the receive data from the IICBnDAT register using the IICBTIAn interrupt.

<6> Data reception completion processing
By clearing the IICBnCTL0.IICBnSLAC bit (to 0) before reading the receive data immediately preceding the final receive data, the next $\overline{\text{ACK}}$ is not output and the end of the data is notified to the transmitting device.

<7> Stop condition output
Set the IICBnTRG.IICBnSPT bit (to 1).

(2) Example of communication in continuous transfer mode (master transmission)



<1> Start condition output
Set the IICBnTRG.IICBnSTT bit (to 1).

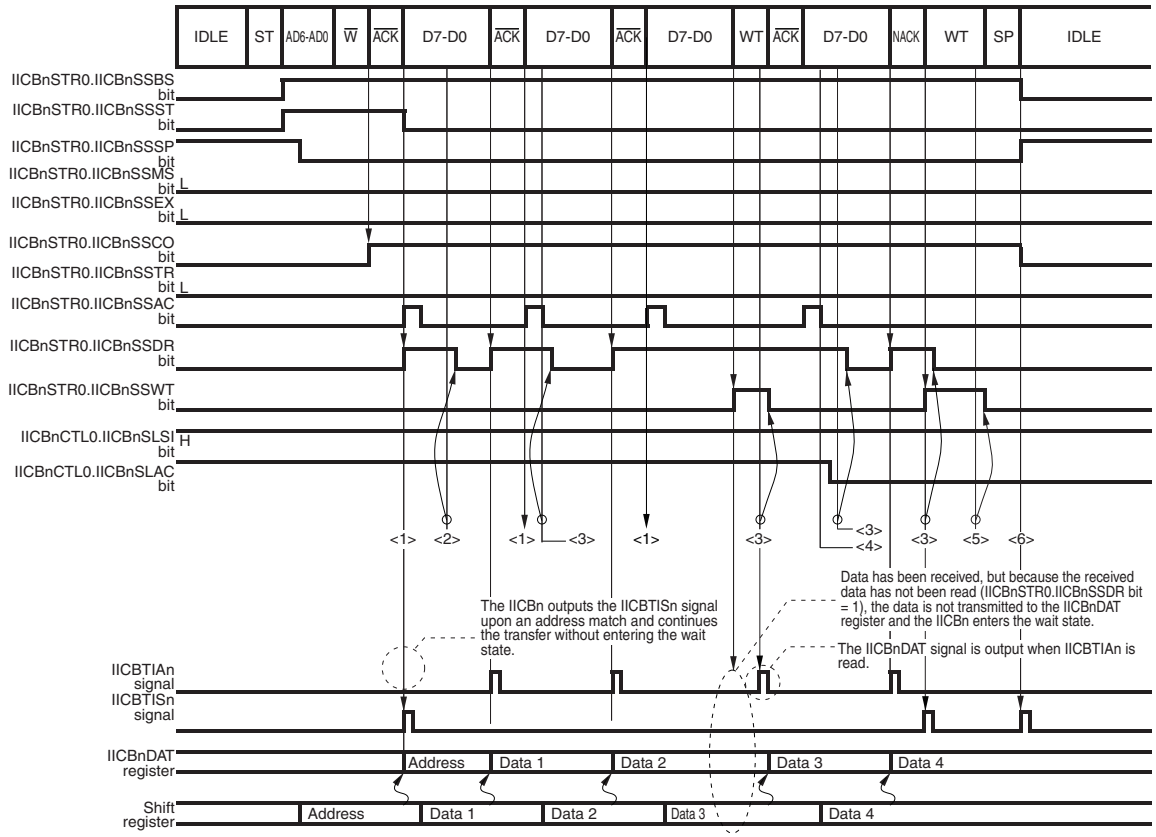
<2> Address and transfer direction specification output
Set the address of the slave device and the transfer direction as 8 bits into the IICBnDAT register.

<3> Data transmission
Set the transmit data to the IICBnDAT register using the IICBTIAn interrupt.

<4> Acknowledge result check
The IICBTISn interrupt occurs only if the slave device does not return $\overline{\text{ACK}}$. Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit.

<5> Stop condition output
Set the IICBnTRG.IICBnSPT bit (to 1).

(3) Example of communication in continuous transfer mode (slave reception)



<1> Data reception
If there is no unread data in the IICBnDAT register by the time reception starts, the IICBn starts reception without entering the wait state.

<2> Operation mode check in slave mode

- Check the operation mode using the IICBTISn interrupt.
- Check the address transfer, address match, and reception operation with the IICBnSTR0.IICBnSSST, IICBnSTR0.IICBnSSCO, and IICBnSTR0.IICBnSSTR bits.
- Read the IICBnDAT register (empty read).

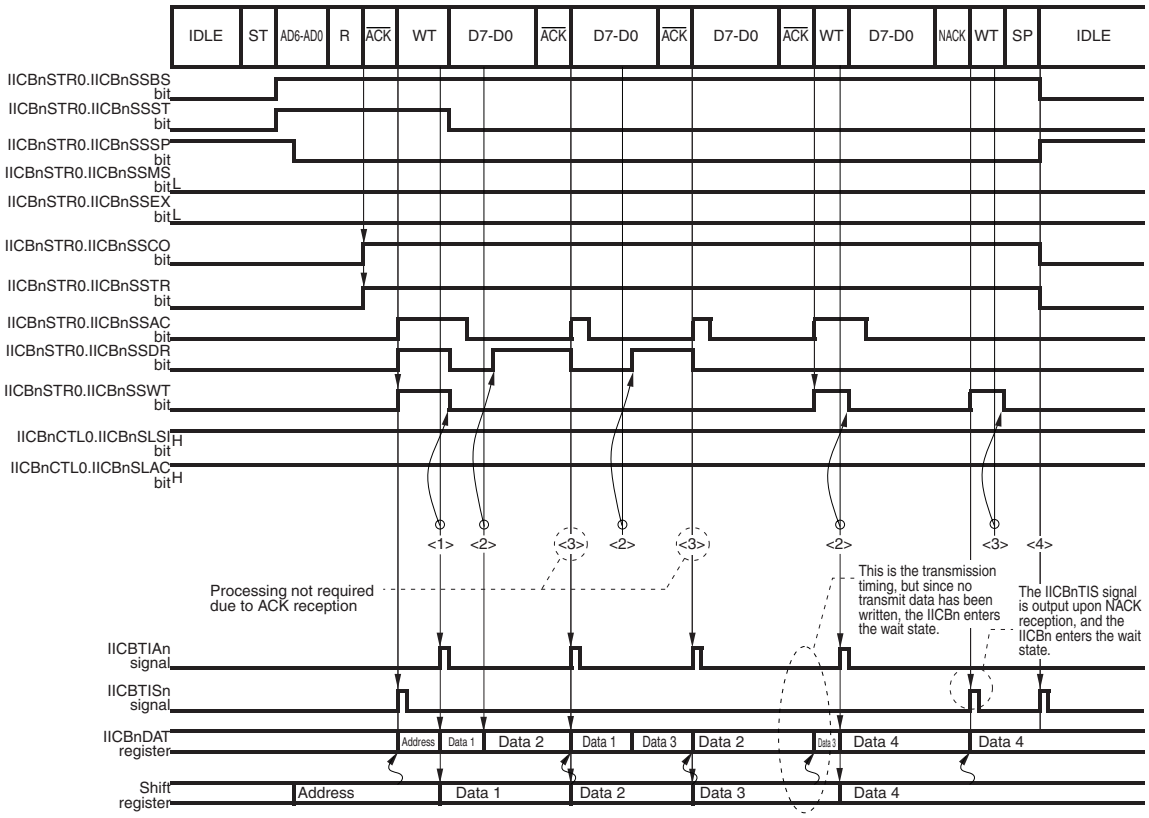
<3> Receive data load
Read the receive data from the IICBnDAT register using the IICBTIAn interrupt.

<4> Data reception completion processing <1>
By clearing the IICBnCTL0.IICBnSLAC bit (to 0) before reading the receive data immediately preceding the final receive data, the next $\overline{\text{ACK}}$ is not output and the end of the data is notified to the transmitting device.

<5> Data reception completion processing <2>
The IICBTISn interrupt occurs only if the slave device does not return $\overline{\text{ACK}}$.
Exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1).

<6> Stop condition detection
Detect the stop condition using the IICBTISn interrupt.

(4) Slave transmission in continuous transfer mode



<1> Operation mode check in slave mode

- Check the operation mode using the IICBTISn interrupt.
- Check the address transfer, address match, and reception operation with the IICBnSTR0.IICBnSSST, IICBnSTR0.IICBnSSCO, and IICBnSTR0.IICBnSSTR bits.
- After reading (empty read) the IICBnDAT register, set the first transmit data to the IICBnDAT register.

<2> Data transmission
Set the transmit data to the IICBnDAT register using the IICBTIAn interrupt.

<3> Acknowledge result check
The IICBTISn interrupt occurs only if the slave device does not return $\overline{\text{ACK}}$. Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit.
If $\overline{\text{ACK}}$ is not output, the transmission is judged to have been completed, and the IICBn exits the wait state by setting the IICBnTRG.IICBnWRET bit (to 1).

<4> Stop condition detection
Detect the stop condition using the IICBTISn interrupt.

26.6.3 Arbitration

When the IICBn operates as the master device and loses arbitration, it enters the slave standby state by setting both SCLn and SDAn to high level upon detection of the arbitration loss, and then the IICBnSTR0.IICBnALDF bit is set (to 1) each time the status interrupt request signal (IICBTISn) is output.

(1) Status upon occurrence of arbitration

The statuses upon occurrence of arbitration during master device operation (IICBnSTR0.IICBnSSMS bit = 1) are listed below.

1. Address transmission
2. R/$\overline{W}$ bit transmission of address transfer
3. Extension code transmission
4. R/$\overline{W}$ bit transmission of extension code transfer
5. Data transmission
6. $\overline{ACK}$ bit transmission after data reception
7. Start condition detection during address transfer or data transfer
8. Stop condition detection during address transfer or data transfer
9. The SDAn signal is low when the IICBn is attempting to output a restart condition
10. The SDAn signal is low when the IICBn is attempting to output a stop condition
11. The falling edge of the SCLn signal is detected when the IICBn is attempting to output a restart condition

26.6.4 Entering and exiting wait state

The IICBn enters the wait state at the following timings.

Table 26-19 Wait state transit timings

Timing	Description	See:
Δ0	Upon detection of the first falling edge of the SCLn, following detection of start condition as the master device	*(1) "Wait state at falling edge of first SCLn after IICBn became master"*
Δ1	Upon detection of the falling edge of the 9th SCLn during address transfer after the start condition	*(2) "Wait state upon detection of the falling edge of the 9th SCLn during address transfer after the start condition"*
Δ2	Upon detection of the falling edge of the 8th SCLn during data transfer	*(3) "Wait state upon detection of the falling edge of the 8th SCLn during data transfer"*
Δ3	Upon detection of the falling edge of the 9th SCLn during data transfer	*(4) "Wait state upon detection of the falling edge of the 9th SCLn during data transfer"*

Note ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
$\overline{ACK}$: Acknowledge
D7 to D0: Data
SP: Stop condition

The method to exit the wait state differs according to the wait state.

Exit the wait state by applying the appropriate method for each of the four wait states as described below.

(1) Wait state at falling edge of first SCLn after IICBn became master

Δ0 indicates the wait state when the data to be transferred has not been written (to the IICBnDAT register) when the falling edge of the first SCLn after the IICBn became the master is detected, after 1 was written to the IICBnTRG.IICBnSTT bit.

(a) Wait state transit condition

The IICBn enters the wait state if data is not written to the IICBnDAT register in the period from when the IICBnTRG.IICBnSTT bit becomes 1 until the Δ0 timing, upon detection of the first falling edge of SCLn after the IICBn became master, after 1 was written to the IICBnTRG.IICBnSTT bit.

However, the valid times to write data to the IICBnDAT register (without entering the wait state) after 1 was written to the IICBnTRG.IICBnSTT bit differ depending on whether the communication reservation function is enabled. The valid times to write to the IICBnDAT register for each of these cases are shown in *Figure 26-11 "Valid times to write to IICBnDAT register"*.

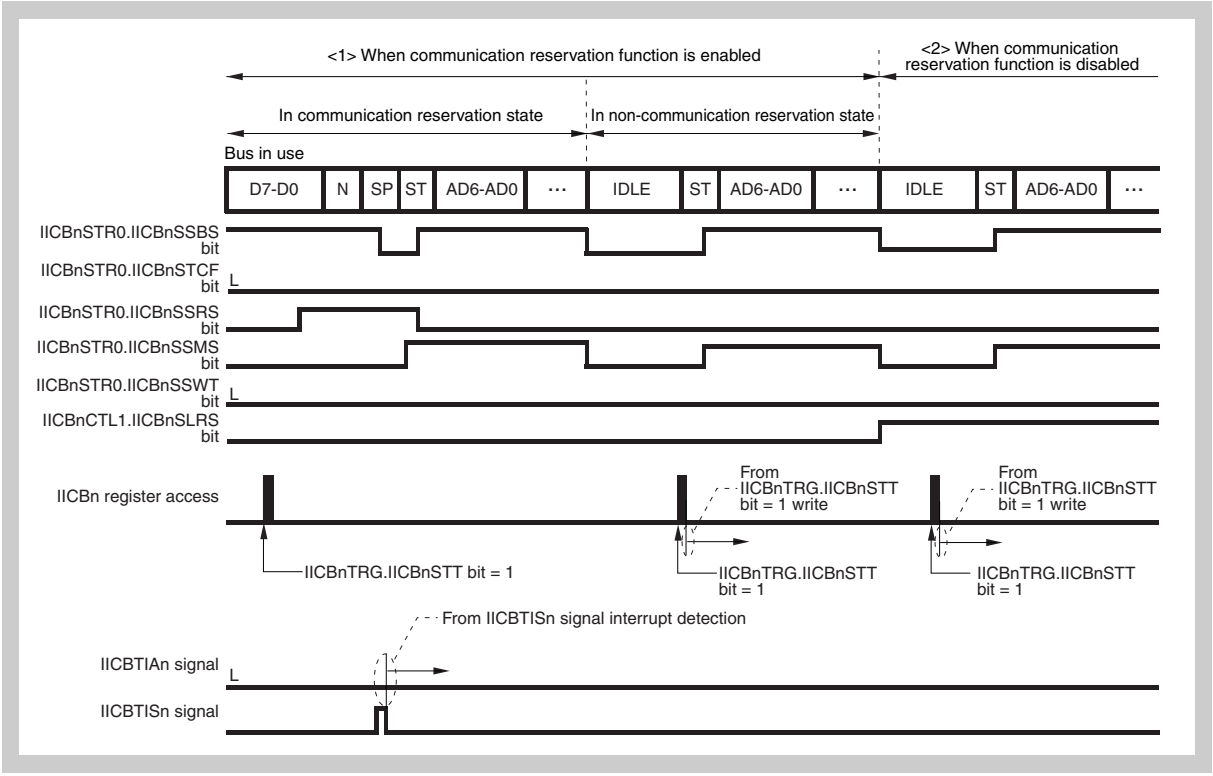


Figure 26-11 Valid times to write to IICBnDAT register

Caution The communication reservation function is disabled (<2> in the above figure) while the IICBnSTR0.IICBnSTCF bit is 0.
When the IICBnSTR0.IICBnSTCF bit becomes 1, setting from IICBnSTR0.IICBnSTCF bit = 1 write is required again.

(b) Wait state exit conditions

Exit the wait state by writing to the IICBnDAT register.

(2) Wait state upon detection of the falling edge of the 9th SCLn during address transfer after the start condition

Δ1 indicates the wait state entered upon completion of address transfer.

(a) Wait state transit condition

<Single transfer mode>

In the single transfer mode, the IICBn always enters the wait state while it operates as the master.

While the IICBn operates as a slave, it enters the wait state upon an address match, or upon extension code detection while the IICBnCTL0.IICBnSLWT bit is 1.

<Continuous transfer mode>

In the continuous transfer mode, the IICBn enters the wait state in the following cases.

- Upon detection of NACK
- When the IICBn operates as the master and transmits data, if the data to be transferred next has not been written
- When the IICBn operates as a slave, if the received data has not been read, or during transmission

(b) Wait state exit conditions

<Single transfer mode>

Exit the wait state by writing to the IICBnDAT register during transmission, or by writing 1 to the IICBnTRG.IICBnWRET bit during reception. When the IICBn operates as the master and the IICBnSTR0.IICBnSSAC bit is 0 or the IICBn is at the transmission side, the wait state can be exited by writing 1 to the IICBnTRG.IICBnSTT or the IICBnTRG.IICBnSPT bit.

<Continuous transfer mode>

Exit the wait state by writing to the IICBnDAT register during transmission, or by reading the IICBnDAT register during reception. When the IICBn operates as the master and the IICBnSTR0.IICBnSSAC bit is 0, the wait state can be exited by writing 1 to the IICBnTRG.IICBnSTT or IICBnTRG.IICBnSPT bit.

(3) Wait state upon detection of the falling edge of the 8th SCLn during data transfer

Δ2 indicates the wait state entered upon detection of the falling edge of the 8th SCLn during data transfer.

(a) Wait state transit condition

<Single transfer mode>

When the IICBn participates in communications and the IICBnCTL0.IICBnSLWT bit is 0, the IICBn enters the wait state if the falling edge of the 8th SCLn is detected.

<Continuous transfer mode>

When the IICBn participates in communications and the IICBnSTR0.IICBnSSTR bit is 0, the IICBn enters the wait state if processing of the previous data (read from the IICBnDAT register) has not completed and 1 has not been written to the IICBnTRG.IICBnSTT and IICBnTRG.IICBnSPT bits before the falling edge of the 8th SCLn.

(b) Wait state exit conditions

<Single transfer mode>

Exit the wait state by writing to the IICBnDAT register during transmission, or by writing 1 to the IICBnTRG.IICBnWRET bit during reception.

<Continuous transfer mode>

Exit the wait state by reading the IICBnDAT register.

(4) Wait state upon detection of the falling edge of the 9th SCLn during data transfer

Δ3 indicates the wait state entered upon detection of the falling edge of the 9th SCLn during data transfer.

During the continuous transfer mode, the IICBn enters the wait state upon NACK reception.

(a) Wait state transit condition

<Single transfer mode>

When the IICBn participates in communications and the IICBnCTL0.IICBnSLWT bit is 1, the IICBn enters the wait state if the falling edge of the 9th SCLn is detected.

<Continuous transfer mode>

When the IICBn participates in communications, it enters the wait state in the following three cases during data transmission:

- Upon reception of NACK by ACK bit while the IICBnCTL0.IICBnSLWT bit is 1
- When transmit data is not written to the data register
- When the previous received data is not read

(b) Wait state exit conditions

The wait state exit conditions are listed for each transfer mode in *Table 26-20 "Wait state exit conditions"*.

Table 26-20 Wait state exit conditions

Master/ slave	Transfer mode	Transfer direction	IICBnSTR0. IICBnSSAC bit	Exit conditions
Master	Single transfer mode	Reception	0	IICBnTRG.IICBnSTT bit = 1 or IICBnTRG.IICBnSPT bit = 1
			1	IICBnTRG.IICBnWRET bit = 1
		Transmission	0	IICBnTRG.IICBnSTT bit = 1 or IICBnTRG.IICBnSPT bit = 1
			1	Write to IICBnDAT register or IICBnTRG.IICBnSTT bit = 1 or IICBnTRG.IICBnSPT bit = 1
	Continuous transfer mode	Reception	0	IICBnTRG.IICBnSTT bit = 1 or IICBnTRG.IICBnSPT bit = 1
			1	Read from IICBnDAT register[a]
		Transmission	0	IICBnTRG.IICBnSTT bit = 1 or IICBnTRG.IICBnSPT bit = 1
			1	Write to IICBnDAT register[b]
Slave	Single transfer mode	Reception	-	IICBnTRG.IICBnWRET bit = 1
		Transmission	0	IICBnTRG.IICBnWRET bit = 1
			1	Write to IICBnDAT register[a]
	Continuous transfer mode	Reception	0	IICBnTRG.IICBnWRET bit = 1
		Transmission	0	IICBnTRG.IICBnWRET bit = 1
			1	Write to IICBnDAT register

a) Condition for exiting the wait state that was entered when no transmit data has been written to the data register
b) Condition for exiting the wait state that was entered when the received data has not been read

26.6.5 Extension code

The processing when the extension code is received differs according to the data after the extension code and thus must be executed through the user's software.

Therefore, the operation differs from that during normal slave address reception. These differences are described below.

(1) When the upper 4 bits of the received address are 0000 or 1111, the extension code reception flag (IICBnSTR0.IICBnSSEX bit) is set to 1 to indicate that an extension code has been received. The status interrupt request signal (IICBTISn) is output at the falling edge of the 8th clock, and the IICBn enters the wait state (IICBnTRG.IICBnSSWT = 1). The IICBnSTR0.IICBnSSDR bit is then set (to 1).

(2) During address transfer, the acknowledge output can be controlled by setting the IICBnCTL0.IICBnSLAC bit. (Note that an acknowledge is always output upon an address match, regardless of the setting of this bit, during address transfer for normal slave address reception.)

(3) The method for exiting the wait state entered upon extension code detection depends on the setting of the IICBnCTL0.IICBnMDTX1 bit as follows.

<When IICBnCTL0.IICBnMDTX1 bit is 0>

During transmission while the IICBnCTL0.IICBnSLWT bit is 0, exit the wait state by writing to the IICBnDAT register. During transmission while the IICBnCTL0.IICBnSLWT bit is 1, or during reception, exit the wait state by writing 1 to the IICBnTRG.IICBnWRET bit.

<When IICBnCTL0.IICBnMDTX1 bit is 1>

During transmission, exit the wait state by writing to the IICBnDAT register, and, during reception, exit the wait state by reading from the IICBnDAT register.

(4) At the falling edge of the 9th clock, if the IICBnCTL0.IICBnSLWT bit is 1, the interrupt request signal (IICBTIAn) is output and the IICBn enters the wait state (IICBnTRG.IICBnSSWT = 1). If the IICBnCTL0.IICBnSLWT bit is 0, the interrupt request signal (IICBTIAn) is not output and the IICBn does not enter the wait state.

(5) If the IICBn receives an extension code, it participates in communications even if the addresses do not match.

For example, to avoid operating the IICBn as a slave device after receiving an extension code, set the IICBnTRG.IICBnLRET bit to 1. The IICBn enters the standby state for the next communication.

26.7 Interrupt Request Signals

Caution In this section, the operation when an extension code is received is omitted. For details about the extension code, see *26.6.5 "Extension code"*.

The IICBn has two interrupt request signals, the data transmit/receive interrupt request signal (IICBTIAn) and the status interrupt request signal (IICBTISn). Both signals are pulses of one PCLK clock width. The interrupt request signal output timing differs according to the transfer mode set using the IICBnCTL0.IICBnMDTX1 and IICBnCTL0.IICBnMDTX0 bits. The interrupt request signals are explained below for each transfer mode.

To perform transfer with an address match between the master device and the slave device, select the single transfer mode or continuous transfer mode with the IICBnCTL0.IICBnMDTX0 bit, and to perform transfer with extension code detection by the slave, select the single transfer mode or continuous transfer mode using the IICBnCTL0.IICBnMDTX1 bit.

26.7.1 Single transfer mode

The interrupt request signal timing in the single transfer mode is described in *Table 26-21 "Interrupt request signal output timing (single transfer mode)"* below.
During the single transfer mode, for the IICBTIAn and IICBTISn interrupt request signals, whether to output an interrupt is judged based on the IICBn state when the falling edge of SCLn is detected during the bus cycle. Note, however, that whether to output an interrupt is judged based on the IICBn state when a stop condition is detected at the Δ4 timing.

Table 26-21 Interrupt request signal output timing (single transfer mode)

Output timing	Description	See:
Δ1	Upon detection of the falling edge of the 9th SCLn during address transfer	*(1) "Interrupt request signal output conditions and output interrupt request signals during address transfer"*
Δ2	Upon detection of the falling edge of the 8th SCLn during data transfer	*(2) "Interrupt request signal output conditions and interrupt request signals output during data transfer"*
Δ3	Upon detection of the falling edge of the 9th SCLn during data transfer	*(2) "Interrupt request signal output conditions and interrupt request signals output during data transfer"*
Δ4	Upon detection of a stop condition	*(3) "Interrupt request signal output upon stop condition detection"*

Note ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
$\overline{ACK}$: Acknowledge
D7 to D0: Data
SP: Stop condition

(1) Interrupt request signal output conditions and output interrupt request signals during address transfer

Δ1 in *Table 26-21 "Interrupt request signal output timing (single transfer mode)"* indicates the interrupt request signal output timing during an address transfer. *Table 26-22 "Interrupt request signal output conditions and interrupt request signals output during address transfer (single transfer mode)"* indicates the interrupt request signal output condition and the interrupt request signal that is output (IICBTIAn or IICBTISn) at the timing of Δ1.

Table 26-22 Interrupt request signal output conditions and interrupt request signals output during address transfer (single transfer mode)

IICBn SSMS	IICBn ALDF	IICBn SLWT	IICBn SSCO	Δ1		Remark
				Interrupt	Wait	
1	0	×	×	IICBTIAn	Wait	-
1	1	×	×	This state does not exist.		-
0	0	×	0	IICBTISn[a]	-	After restart, non-participation in communications
0	0	×	1	IICBTIAn	Wait	-
0	1	×	0	IICBTISn	-	After arbitration loss, non-participation in communications
0	1	×	1	IICBTIAn	Wait	-

a) In case of an address match or extension code detection, before the restart condition

Note ×: don't care

(2) Interrupt request signal output conditions and interrupt request signals output during data transfer

Δ2 and Δ3 in *Table 26-21 "Interrupt request signal output timing (single transfer mode)"* indicate the interrupt request signal output timing during a data transfer. The interrupt request signal output timing of Δ2 or Δ3 is determined according to the setting of the IICBnCTL0.IICBnSLWT bit. *Table 26-23 "Interrupt request signal output conditions and interrupt request signals output during address transfer (single transfer mode)"* indicates the interrupt request signal output condition and the interrupt request signal that is output (IICBTIAn or IICBTISn) at the timing of Δ2 and Δ3.

Table 26-23 Interrupt request signal output conditions and interrupt request signals output during address transfer (single transfer mode)

IICBn SSMS	IICBn ALDF	IICBn SLWT	IICBn SSCO	Δ2		Δ3		Remark
				Interrupt	Wait	Interrupt	Wait	
1	0	0	×	IICBTIAn	Wait	-	-	-
1	0	1	×	-	-	IICBTIAn	Wait	-
1	1	×	×	This state does not exist.				-
0	0	×	0	-	-	-	-	Non-participation in communications
0	0	0	1	IICBTIAn	Wait	-	-	-
0	0	1	1	-	-	IICBTIAn	Wait	-
0	1	0	0	IICBTISn	-	-	-	Non-participation in communications after arbitration loss
0	1	1	0	-	-	IICBTISn	-	Non-participation in communications after arbitration loss
0	1	0	1	IICBTIAn	Wait	-	-	-
0	1	1	1	-	-	IICBTIAn	Wait	-

Note ×: don't care

(3) Interrupt request signal output upon stop condition detection

Δ4 in *Table 26-21 "Interrupt request signal output timing (single transfer mode)"* indicates the interrupt request signal output timing upon detection of a stop condition.

Interrupt request signal output is controlled according to the IICBnCTL0.IICBnSLSI bit. If a stop condition is detected while the IICBnCTL0.IICBnSLSI bit is 1, the status interrupt request signal (IICBTISn) is output.

26.7.2 Continuous transfer mode

(1) Data transmit/receive interrupt request signal (IICBTIAn)

The conditions for outputting an IICBTIAn signal in the continuous transfer mode are described below.

- Interrupt request signal output condition during reception

  When receive data is saved from the shift register to the IICBnDAT register (timing <1> in *Figure 26-12 "IICBTIAn signal output timing (reception in continuous transfer mode)"*)

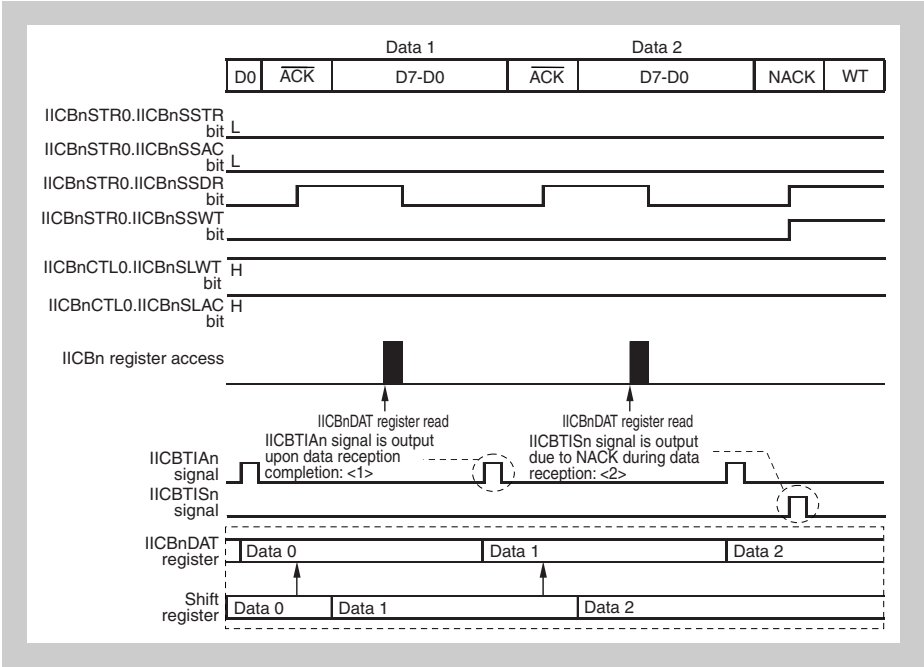


Figure 26-12 IICBTIAn signal output timing (reception in continuous transfer mode)

- Interrupt request signal output condition during transmission

  When data is written to the IICBnDAT register while there is no transmit data in the shift register and IICBnDAT register (timing <2> in *Figure 26-13 "IICBTIAn signal output timing (transmission in continuous transfer mode)"*).

  When data is saved from the IICBnDAT register to the shift register (timing <1> in *Figure 26-13 "IICBTIAn signal output timing (transmission in continuous transfer mode)"*).

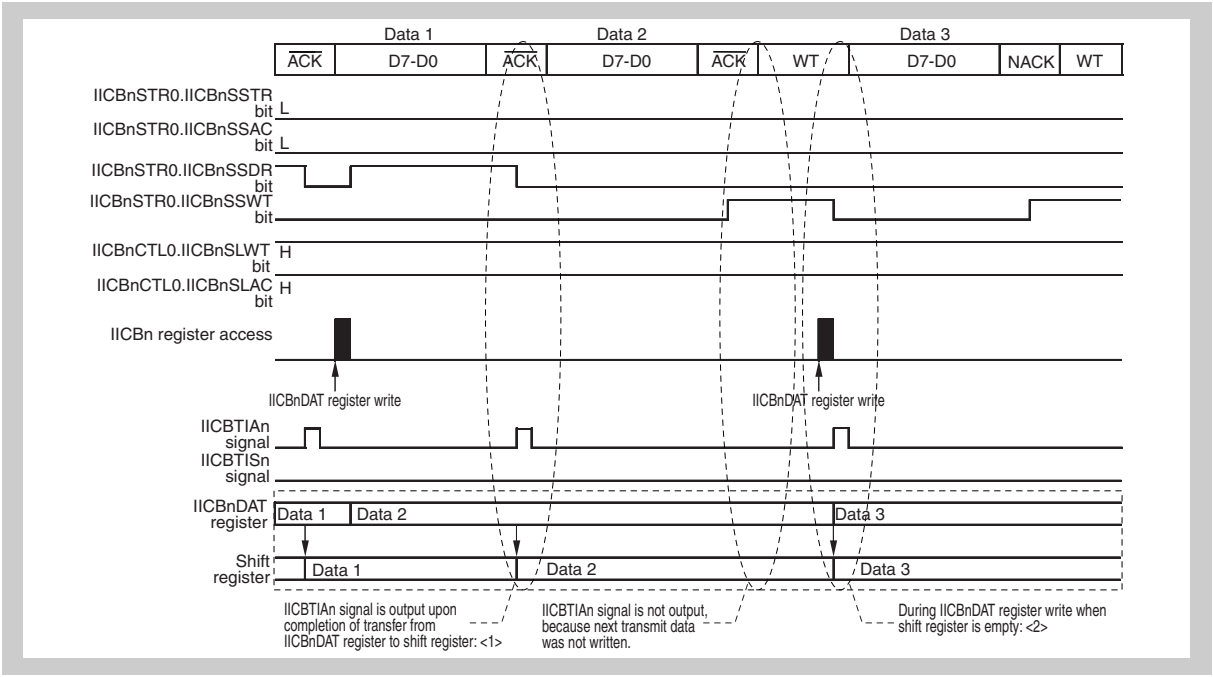


Figure 26-13 IICBTIAn signal output timing (transmission in continuous transfer mode)

(2) Status interrupt request signal (IICBTISn)

The IICBTISn signal output timing in the continuous transfer mode is the same as that in the single transfer mode.

Table 26-24 IICBTISn signal output timing

Output timing	Description	See:
Δ1	Upon detection of the falling edge of the 9th SCLn during address transfer after the start condition	*(a) "IICBTISn signal output conditions during address transfer"*
Δ2	Upon detection of the falling edge of the 8th SCLn during data transfer	*(b) "IICBTISn signal output conditions during data transfer"*
Δ3	Upon detection of the falling edge of the 9th SCLn during data transfer	*(b) "IICBTISn signal output conditions during data transfer"*
Δ4	Upon detection of a stop condition	*(c) "IICBTISn signal output upon detection of stop condition"*

Note ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
$\overline{ACK}$: Acknowledge
D7 to D0: Data
SP: Stop condition

(a) IICBTISn signal output conditions during address transfer

Δ1 in *Table 26-24 "IICBTISn signal output timing"* indicates the IICBTISn signal output timing during address transfer. *Table 26-25 "IICBTISn signal output conditions during address transfer (continuous transfer mode)"* indicates the IICBTISn signal output conditions at the Δ1 timing.

Table 26-25 IICBTISn signal output conditions during address transfer (continuous transfer mode)

IICBn SSMS	IICBn SSCO	IICBn ALDF	Transfer direction	IICBn SSDR	IICBn SSAC	Δ1	
						Interrupt	Wait
1	×	0	Transmission	0	1	-	Wait
1	×	0	Transmission	0	0	IICBTISn	Wait
1	×	0	Transmission	1	1	-	-
1	×	0	Transmission	1	0	IICBTISn	Wait
1	×	0	Reception	0	1	-	-
1	×	0	Reception	0	0	IICBTISn	Wait
1	×	0	Reception	1	1	IICBTISn during IICBnDAT read[a]	Wait
1	×	0	Reception	1	0	IICBTISn during IICBnDAT read	Wait
1	×	1	×	×	×	This state does not exist.	
0	0	0	×	×	×	IICBTISn[b]	-
0	0	1	×	×	×	IICBTISn	-
0	1	×	Transmission	×	1	IICBTISn	Wait
0	1	×	Reception	0	1	IICBTISn	-
0	1	×	Reception	1	1	IICBTISn during IICBnDAT read	Wait

a) Upon restarting without reading IICBnDAT after the reception ends
b) Upon an address match before restart condition

Caution For Δ1, the IICBnSTR0.IICBnSSAC bit is always 0.

Note ×: don't care

(b) IICBTISn signal output conditions during data transfer

Δ2 and Δ3 in *Table 26-24 "IICBTISn signal output timing"* indicate the IICBTISn signal output timings during data transfer. *Table 26-26 "IICBTISn signal output conditions during data transfer (continuous transfer mode)"* indicates the IICBTISn signal output conditions at the Δ2 and Δ3 timings.

Table 26-26 IICBTISn signal output conditions during data transfer (continuous transfer mode)

IICBn SSMS	IICBn SSCO	IICBn SLWT	IICBn ALDF	Transfer direction	IICBn SSDR	IICBn SSAC	IICBnSTT or IICBnSPT	Δ2		Δ3	
								Interrupt	Wait	Interrupt	Wait
1	×	0	×	Transmission	0	1	a	-	-	-	Wait
1	×	0	×	Transmission	0	0	a	-	-	IICBTISn	Wait
1	×	0	×	Transmission	1	1	a	-	-	-	-
1	×	0	×	Transmission	1	0	a	-	-	IICBTISn	Wait
1	×	0	×	Reception	0	1	a	-	-	-	-
1	×	0	×	Reception	0	0	a	-	-	IICBTISn	Wait
1	×	0	×	Reception	1	1	a	-	-	-	-
1	×	0	×	Reception	1	0	a	-	-	IICBnDAT after IICBTISn read	Wait
1	×	×	×	×	×	0	b		-	IICBTISn	-
1	×	×	×	×	×	1	b	-	-	-	-
0	0	×	0	×	×	×	×	-	-	-	-
0	0	0	1	Reception	×	×	×	IICBTIS	-	-	-
0	0	1	1	Transmission	×	×	×	-	-	IICBTISn	-
0	1	0	×	Transmission	0	1	a	-	-	-	Wait
0	1	0	×	Transmission	0	0	a	-	-	IICBTISn	Wait
0	1	0	×	Transmission	1	1	a	-	-	-	-
0	1	0	×	Transmission	1	0	a	-	-	IICBTISn	Wait
0	1	0	×	Reception	0	1	a	-	-	-	-
0	1	0	×	Reception	0	0	a	-	-	IICBTISn	Wait
0	1	0	×	Reception	1	1	a	-	-	-	-
0	1	0	×	Reception	1	0	a	-	-	IICBnDAT during IICBTISn read	Wait

a) When 1 has not been written to the IICBnTRG.IICBnSTT or IICBnTRG.IICBnSPT bit
b) When 1 has been written to the IICBnTRG.IICBnSTT or IICBnTRG.IICBnSPT bit

Note ×: don't care

(c) IICBTISn signal output upon detection of stop condition

∆4 in *Table 26-24 "IICBTISn signal output timing"* indicates the IICBTISn signal output timing upon detection of a stop condition.

IICBTISn signal output is controlled according to the IICBnCTL0.IICBnSLSI bit. If a stop condition is detected while the IICBnCTL0.IICBnSLSI bit is 1, the IICBTISn signal is output.

26.8 Interrupt Outputs and Statuses

This section describes the statuses of the IICBnSTR0 register during interrupt output by communication flow.

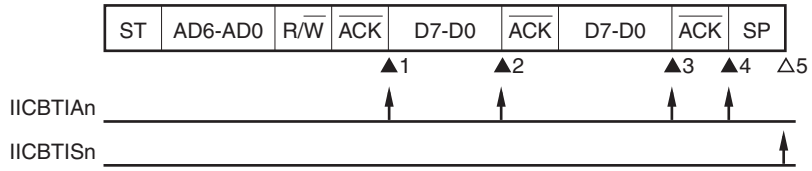
The meanings of the symbols used in the figures are as follows.

ST:	Start condition
AD6-AD0:	Address
R, $\overline{W}$, R/$\overline{W}$:	Transfer direction specification
$\overline{ACK}$:	Acknowledge
NACK:	Not acknowledge
D7-D0:	Data
SP:	Stop condition

26.8.1 Single transfer mode (master device operation)

(1) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-0100X1 0110--00B

▲2: IICBnSTR0 register = 1-0100X0 0100--00B

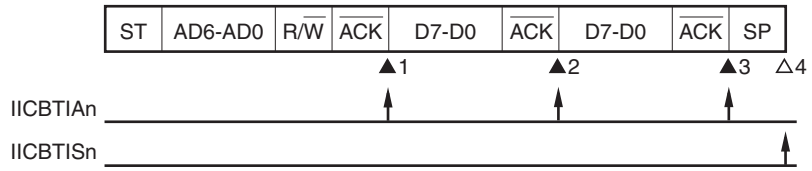
▲3: IICBnSTR0 register = 1-0100X0 0100--00B (IICBnCTL0.IICBnSLWT bit = 1)

▲4: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnTRG.IICBnSPT bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-0100X1 0110--00B

▲2: IICBnSTR0 register = 1-0100X0 0100--00B

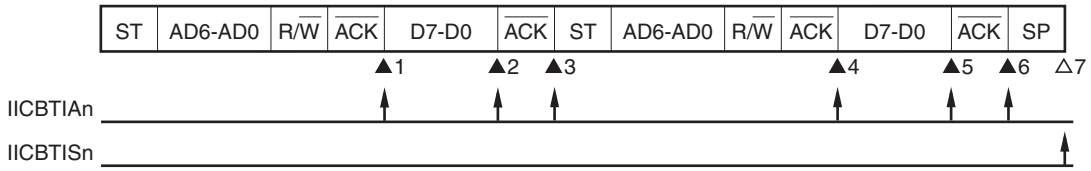
▲3: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnTRG.IICBnSPT bit = 1)

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-0100X1 0110--00B

▲2: IICBnSTR0 register = 1-0100X0 0100--00B (IICBnCTL0.IICBnSLWT bit = 1)

▲3: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnTRG.IICBnSTT bit = 1, IICBnCTL0.IICBnSLWT bit = 0)

▲4: IICBnSTR0 register = 1-0100X1 0110--00B

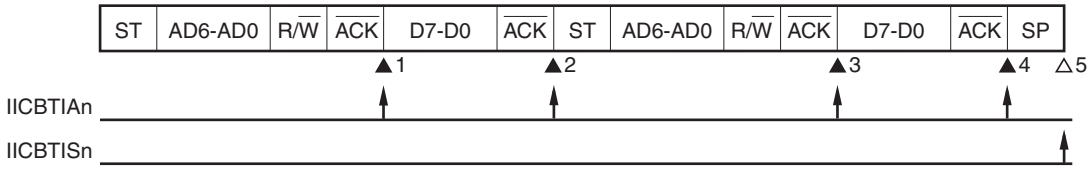
▲5: IICBnSTR0 register = 1-0100X0 0100--00B (IICBnCTL0.IICBnSLWT bit = 1)

▲6: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnTRG.IICBnSPT bit = 1)

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-0100X1 0110--00B

▲2: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnTRG.IICBnSTT bit = 1)

▲3: IICBnSTR0 register = 1-0100X1 0110--00B

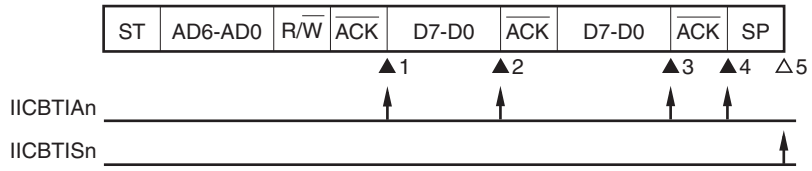
▲4: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnTRG.IICBnSPT bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

(3) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-0110X1 0110--00B

▲2: IICBnSTR0 register = 1-0110X0 0100--00B

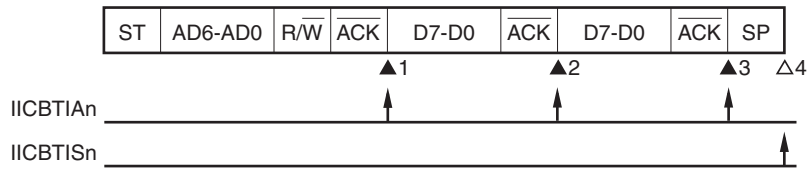
▲3: IICBnSTR0 register = 1-0110X0 0100--00B (IICBnCTL0.IICBnSLWT bit = 1)

▲4: IICBnSTR0 register = 1-0110XX 0100--00B (IICBnTRG.IICBnSPT bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-0110X1 0110--00B

▲2: IICBnSTR0 register = 1-0110X1 0100--00B

▲3: IICBnSTR0 register = 1-0110XX 0100--00B (IICBnTRG.IICBnSPT bit = 1)

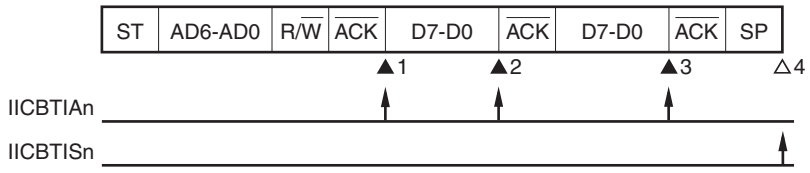
△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

26.8.2 Single transfer mode (slave device operation: during slave address reception (IICBnSTR0.IICBnSSC0 bit = 1))

(1) Start ~ Address ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

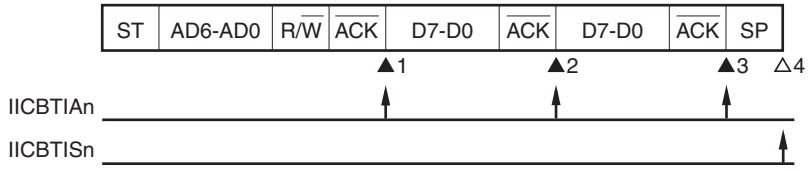
▲2: IICBnSTR0 register = 0-0101X0 0100--00B

▲3: IICBnSTR0 register = 0-0101X0 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101X1 0100--00B

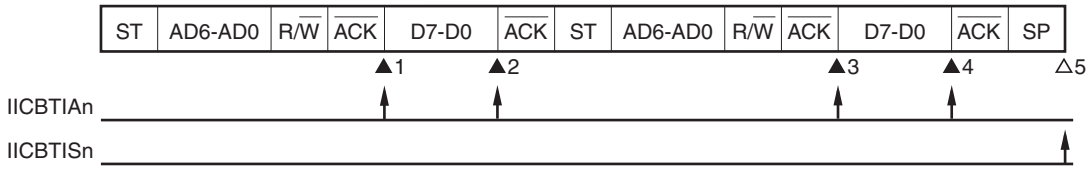
▲3: IICBnSTR0 register = 0-0101XX 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, address match)



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101X0 0100--00B

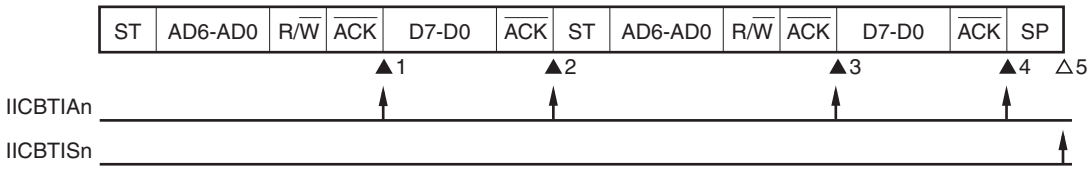
▲3: IICBnSTR0 register = 0-0101X1 0110--00B

▲4: IICBnSTR0 register = 0-0101X0 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, address match)



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101XX 0100--00B

▲3: IICBnSTR0 register = 0-0101X1 0110--00B

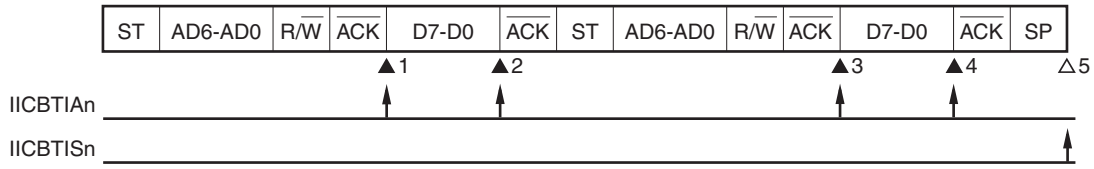
▲4: IICBnSTR0 register = 0-0101XX 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101X0 0100--00B

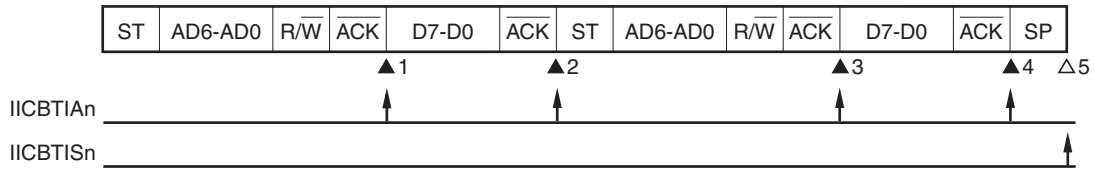
▲3: IICBnSTR0 register = 0-0110X1 0110--00B

▲4: IICBnSTR0 register = 0-0110X0 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101XX 0100--00B

▲3: IICBnSTR0 register = 0-0110X1 0110--00B

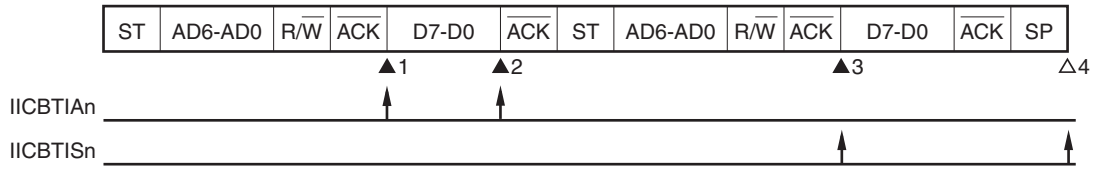
▲4: IICBnSTR0 register = 0-0110XX 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

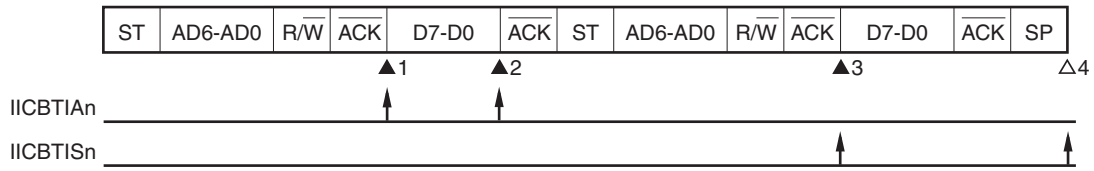
▲2: IICBnSTR0 register = 0-0101X0 0100--00B

▲3: IICBnSTR0 register = 0-0000X0 0110--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101X0 0100--00B

▲3: IICBnSTR0 register = 0-0000X0 0110--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

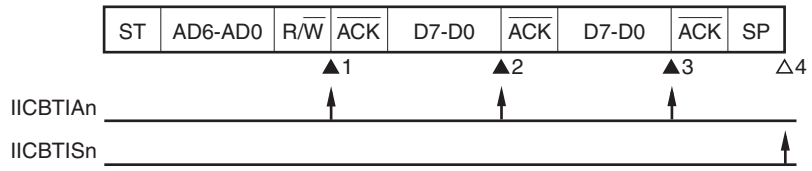
Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

26.8.3 Single transfer mode (slave device operation: during extension code reception (IICBnSTR0.IICBnSSEX bit = 1))

The IICBn always participates in communications when it receives an extension code.

(1) Start ~ Code ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

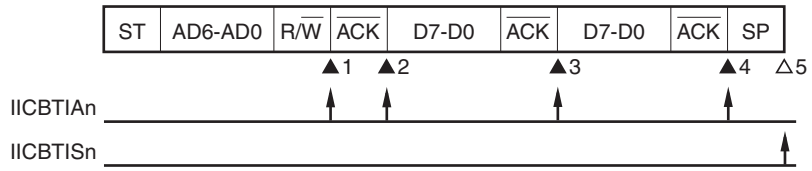
▲2: IICBnSTR0 register = 0-0110X0 0100--00B

▲3: IICBnSTR0 register = 0-0110X0 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0110X0 0100--00B

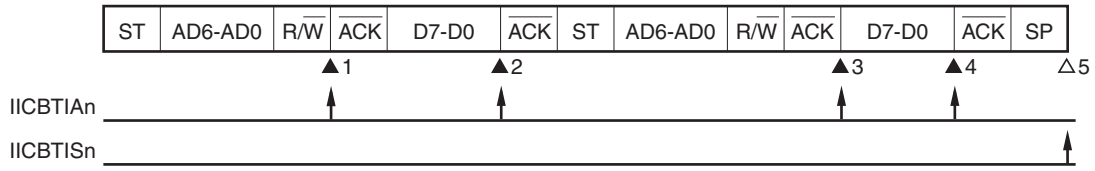
▲4: IICBnSTR0 register = 0-0110XX 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, address match)



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X0 0100--00B

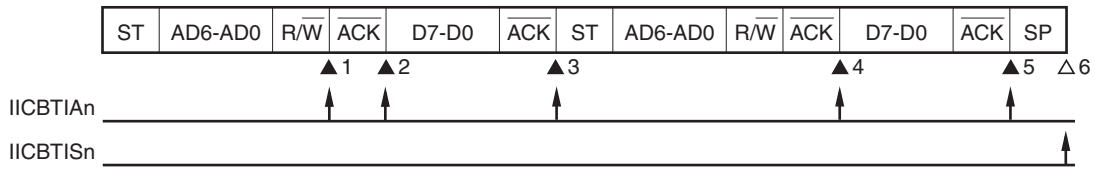
▲3: IICBnSTR0 register = 0-0101X1 0110--00B

▲4: IICBnSTR0 register = 0-0101X0 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, address match)



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0110X0 0100--00B

▲4: IICBnSTR0 register = 0-0101X1 0110--00B

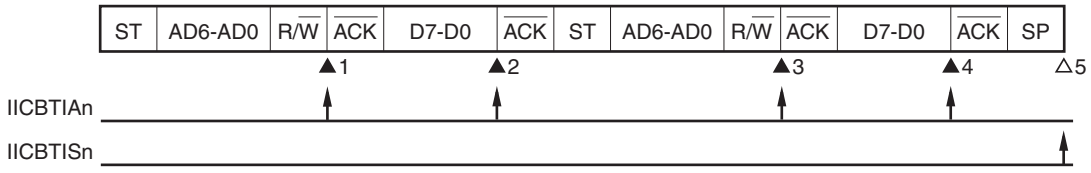
▲5: IICBnSTR0 register = 0-0101XX 0100--00B

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X0 0100--00B

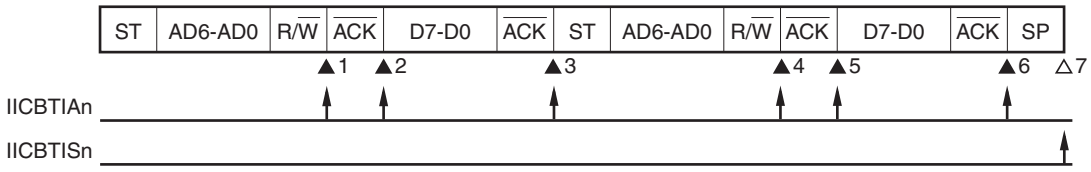
▲3: IICBnSTR0 register = 0-0110X0 0110--00B

▲4: IICBnSTR0 register = 0-0110X0 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0110XX 0100--00B

▲4: IICBnSTR0 register = 0-0110X0 0110--00B

▲5: IICBnSTR0 register = 0-0110X1 0110--00B

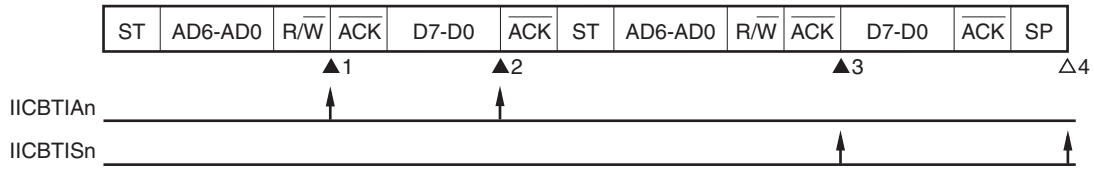
▲6: IICBnSTR0 register = 0-0110XX 0100--00B

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

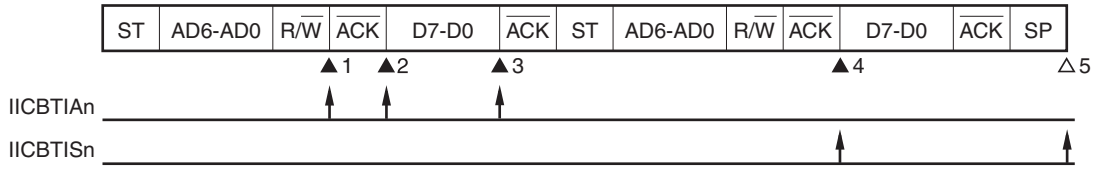
▲2: IICBnSTR0 register = 0-0110X0 0100--00B

▲3: IICBnSTR0 register = 0-0000X0 0110--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0000X0 0100--00B

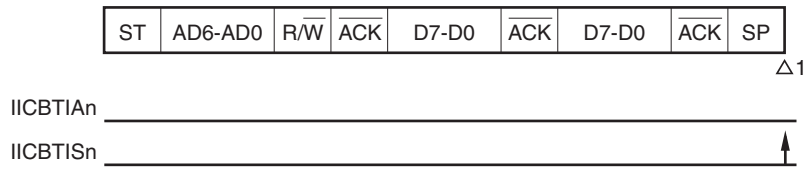
▲4: IICBnSTR0 register = 0-0000X0 0110--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

26.8.4 Single transfer mode (non-participation in communications)

(1) Start ~ Code ~ Data ~ Data ~ Stop



△1: IICBnSTR0 register = 0-000000 0001--00B

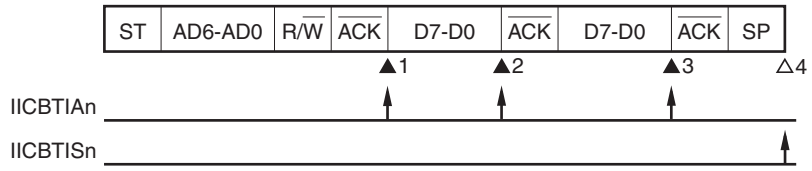
Note △ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

26.8.5 Single transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1): operation as slave after arbitration loss)

When using IICBn as the master in a multi-master system, read the IICBnSTR0.IICBnALDF bit for each IICBTISn interrupt occurrence to confirm the arbitration result.

(1) Address match after arbitration loss

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 0-0101X1 0110--01B (IICBnSTRC.IICBnCLAF bit = 1)

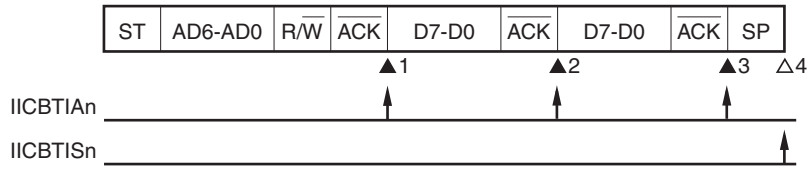
▲2: IICBnSTR0 register = 0-0101X0 0100--00B

▲3: IICBnSTR0 register = 0-0101X0 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-0101X1 0110--01B (IICBnSTRC.IICBnCLAF bit = 1)

▲2: IICBnSTR0 register = 0-0101X1 0100--00B

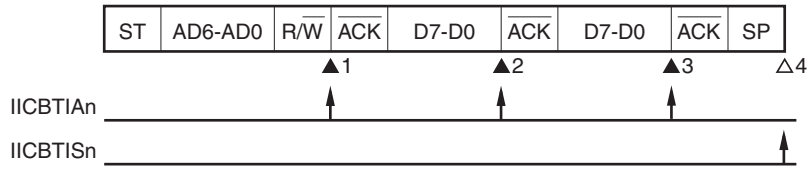
▲3: IICBnSTR0 register = 0-0101XX 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(2) Upon extension code detection after arbitration loss

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 0-0110X0 0110--01B (IICBnSTRC.IICBnCLAF bit = 1)

▲2: IICBnSTR0 register = 0-0110X0 0100--00B

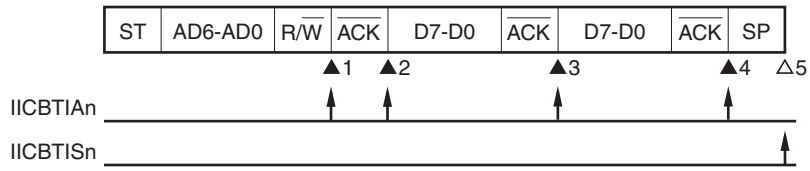
▲3: IICBnSTR0 register = 0-0110X0 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Notes 1. ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

2.n = 0 to 5

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-0110X0 0110--01B (IICBnSTRC.IICBnCLAF bit = 1)

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0110X0 0100--00B

▲4: IICBnSTR0 register = 0-0110XX 0100--00B

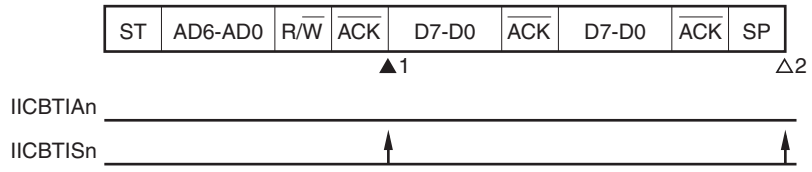
△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

26.8.6 Single transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1): non-participation in communications after arbitration loss)

When using IICBn as the master in a multi-master system, read the IICBnSTR0.IICBnALDF bit for each IICBTISn interrupt occurrence to confirm the arbitration result.

(1) Arbitration loss during transmission of slave address



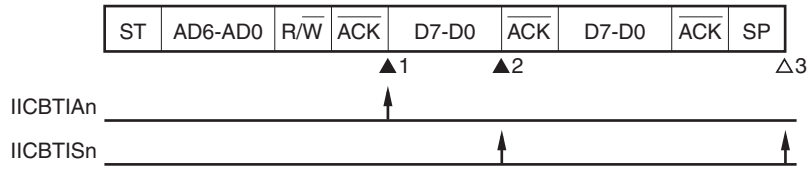
▲1: IICBnSTR0 register = 0-0000X1 0110--01B (IICBnSTRC.IICBnCLAF bit = 1)

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don’t care

(2) Arbitration loss during data transfer

(a) When IICBnCTL0.IICBnSLWT bit is 0



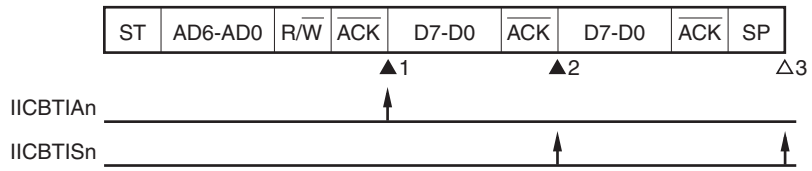
▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnSTRC.IICBnCLAF bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

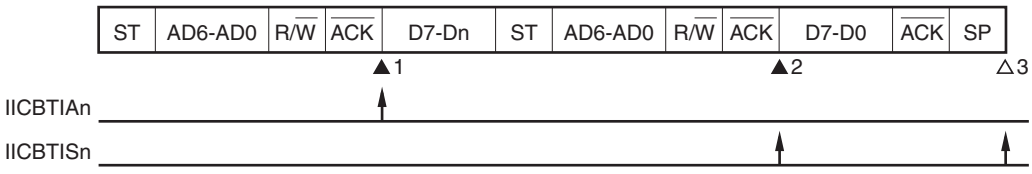
▲2: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnSTRC.IICBnCLAF bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(3) Arbitration loss for the restart condition during data transfer

(a) When IICBnCTL0.IICBnSLWT bit is 1 (extension code mismatch, address mismatch)



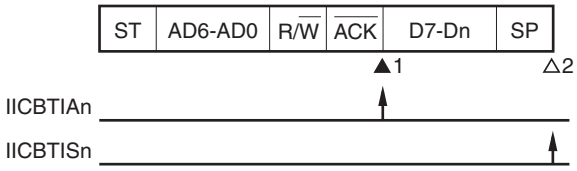
▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnSTRC.IICBnCLAF bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(4) Arbitration loss for the stop condition during data transfer



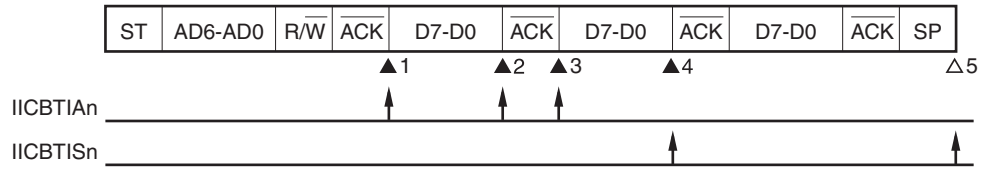
▲1: IICBnSTR0 register = 1-1000X1 0110--00B

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output regardless of the setting of IICBnCTL0.IICBnSLSI bit
\- Undefined
X don't care

(5) Arbitration loss because the SDAn signal is low level when attempting to output restart condition

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 1-1000X0 0100--00B (IICBnCTL0.IICBnSLWT bit = 1)

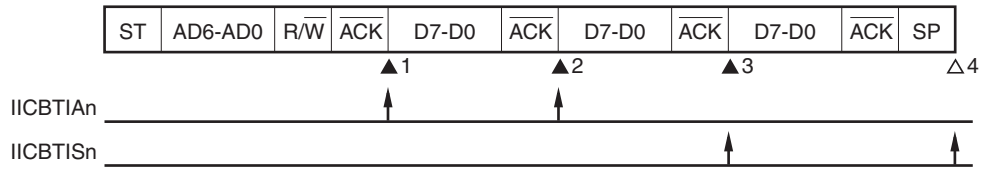
▲3: IICBnSTR0 register = 1-1000XX 0100--00B (IICBnCTL0.IICBnSLWT bit = 0, IICBnTRG.IICBnSTT bit = 1)

▲4: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnSTRC.IICBnCLAF bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 1-1000XX 0100--00B (IICBnCTL0.IICBnSLWT bit = 0, IICBnTRG.IICBnSTT bit = 1)

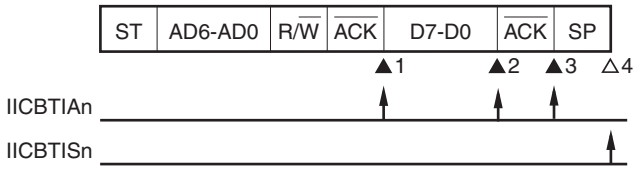
▲3: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnSTRC.IICBnCLAF bit = 1)

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(6) Arbitration loss for the stop condition when attempting to output restart condition

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

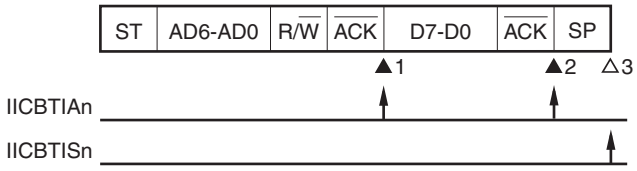
▲2: IICBnSTR0 register = 1-1000X0 0100--00B (IICBnCTL0.IICBnSLWT bit = 0)

▲3: IICBnSTR0 register = 1-0000XX 0100--00B (IICBnTRG.IICBnSTT bit = 1)

△4: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output regardless of the setting of IICBnCTL0.IICBnSLSI bit
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

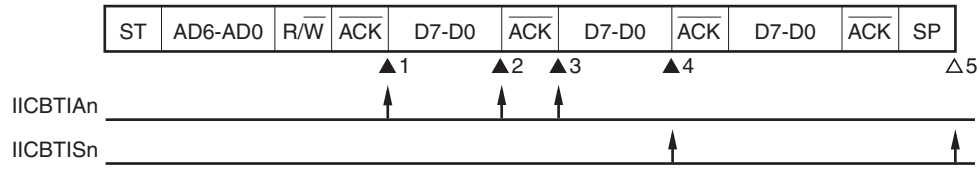
▲2: IICBnSTR0 register = 1-0000XX 0100--00B (IICBnTRG.IICBnSTT bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output regardless of the setting of IICBnCTL0.IICBnSLSI bit
\- Undefined
X don't care

(7) Arbitration loss because the SDAn signal is low level when attempting to output stop condition

(a) When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 1-1000X0 0100--00B (IICBnCTL0.IICBnSLWT bit = 1)

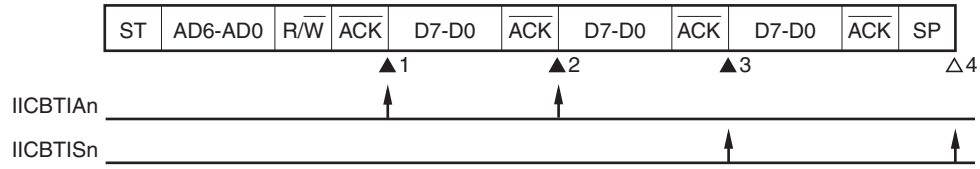
▲3: IICBnSTR0 register = 1-1000XX 0100--00B (IICBnCTL0.IICBnSLWT bit = 0, ICBnTRG.IICBnSPT bit = 1)

▲4: IICBnSTR0 register = 0-0000XX 0100--01B (IICBnSTRC.IICBnCLAF bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(b) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 1-1000XX 0100--00B (IICBnTRG.IICBnSPT bit = 1)

▲3: IICBnSTR0 register = 0-0000XX 0100--01B (IICBnSTRC.IICBnCLAF bit = 1)

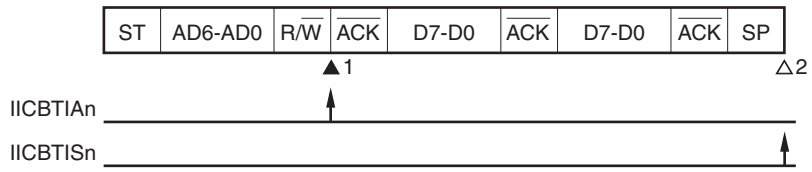
△4: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

26.8.7 Single transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1): non-participation in communications after arbitration loss (during extension code transfer))

When using IICBn as the master in a multi-master system, read the IICBnSTR0.IICBnALDF bit for each IICBTISn interrupt occurrence to confirm the arbitration result.

(1) Arbitration loss during extension code transfer

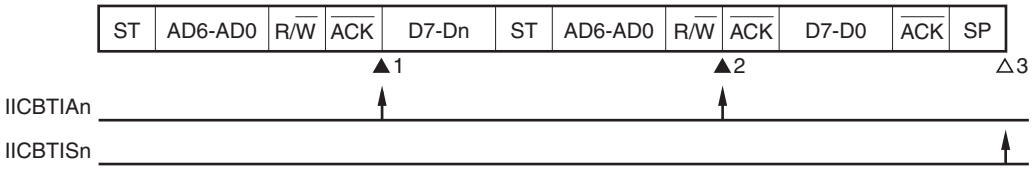


▲1: IICBnSTR0 register = 0-1100X0 0110--01B (IICBnSTRC.IICBnCLAF bit = 1, IICBnTRG.IICBnLRET bit = 1)

△2: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(2) Arbitration loss for the restart condition during data transfer (extension code match)



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 0-1100X0 0100--01B (IICBnSTRC.IICBnCLAF bit = 1, IICBnTRG.IICBnLRET bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--01B

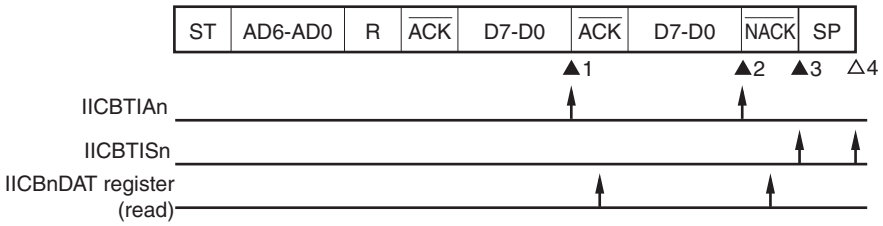
Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

26.8.8 Continuous transfer mode (master device operation (reception))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Address ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 1-100000 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲2: IICBnSTR0 register = 1-100000 0100--00B]

IICBnDAT register read

→ IICBnSTR0 register = 1-000000 0100--00B

▲3: IICBnSTR0 register = 1-010000 0100--00B

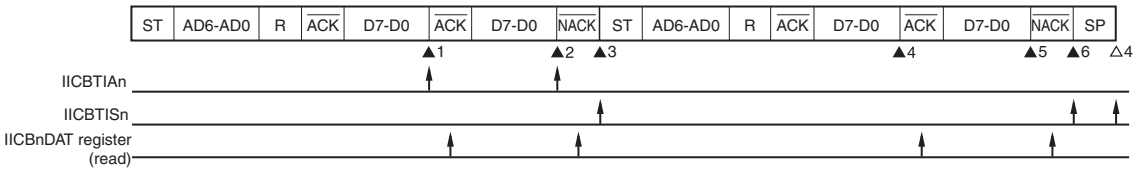
→ IICBnTRG.IICBnSPT bit = 1

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined

(2) Start ~ Address ~ Data × 2 ~ Start ~ Address ~ Data × 2 ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 1-100001 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲2: IICBnSTR0 register = 1-100000 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 1-010000 0100--00B

▲3: IICBnSTR0 register = 1-010000 0100--00B

→ IICBnTRG.IICBnSTT bit = 1

[▲4: IICBnSTR0 register = 1-100000 0100--00B]

IICBnDAT register read

[▲5: IICBnSTR0 register = 1-100000 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 1-000000 0100--00B

▲6: IICBnSTR0 register = 1-010000 0100--00B

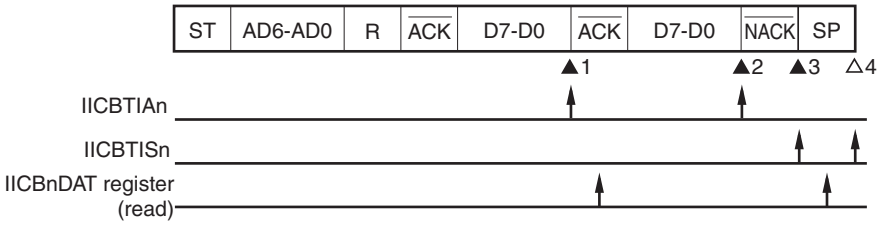
→ IICBnTRG.IICBnSTT bit = 1

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined

(3) Start ~ Code ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 1-101001 0100--00B]

IICBnDAT register read

→ IICBnSTR0 register = 1-0010001 0100--00B

[▲2: IICBnSTR0 register = 1-101000 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 1-011000 0100--00B

▲3: IICBnSTR0 register = 1-01000 0100--00B

→ IICBnTRG.IICBnSPT bit = 1

△4: IICBnTRG.IICBnSTR0 register = 0-000000 0001--00B

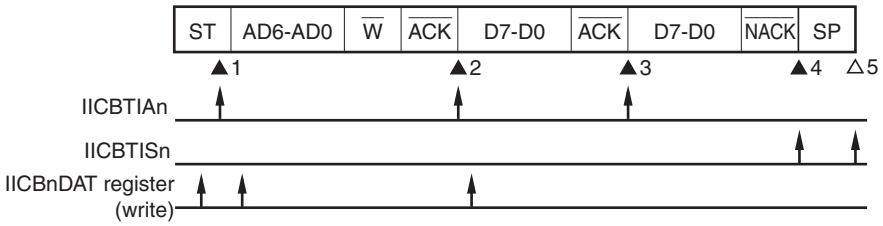
Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

26.8.9 Continuous transfer mode (master device operation (transmission))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Address ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1



IICBnDAT register write (address)

[▲1: IICBnSTR0 register = X-0000X0 0X0X--00B]

IICBnDAT register write

[▲2: IICBnSTR0 register = 1-000011 0110--00B]

IICBnDAT register write

[▲3: IICBnSTR0 register = 1-000011 0100--00B]

▲4: IICBnSTR0 register = 1-010010 0100--00B

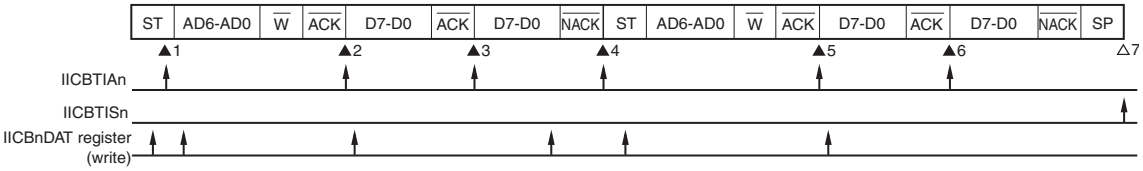
IICBnTRG.IICBnSPT bit = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

(2) Start ~ Address ~ Data × 2 ~ Start ~ Address ~ Data × 2 ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1



IICBnDAT register write (address)

[▲1: IICBnSTR0 register = X-0000X0 0X0X--00B]

IICBnDAT register write

[▲2: IICBnSTR0 register = 1-000011 0110--00B]

IICBnDAT register write

[▲3: IICBnSTR0 register = 1-000011 0100--00B]

IICBnTRG.IICBnSTT bit = 1

IICBnDAT register write (address)

[▲4: IICBnSTR0 register = 1-000010 010X--00B]

IICBnDAT register write

[▲5: IICBnSTR0 register = 1-000011 0110--00B]

IICBnDAT register write

[▲6: IICBnSTR0 register = 1-000011 0110--00B]

IICBnTRG.IICBnSPT bit = 1

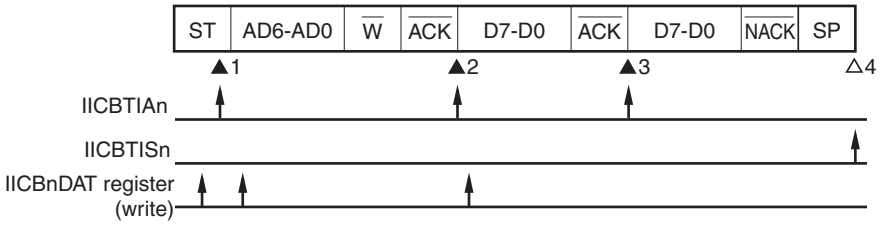
IICBnDAT register write

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

(3) Start ~ Code ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1



IICBnDAT register write (address)

[▲1: IICBnSTR0 register = X-0000X0 0X0X--00B]

IICBnDAT register write

[▲2: IICBnSTR0 register = 1-000011 0110--00B]

IICBnDAT register write

[▲3: IICBnSTR0 register = 1-000011 0100--00B]

IICBnTRG.IICBnSPT bit = 1

△4: IICBnSTR0 register = 0-000000 0001--00B

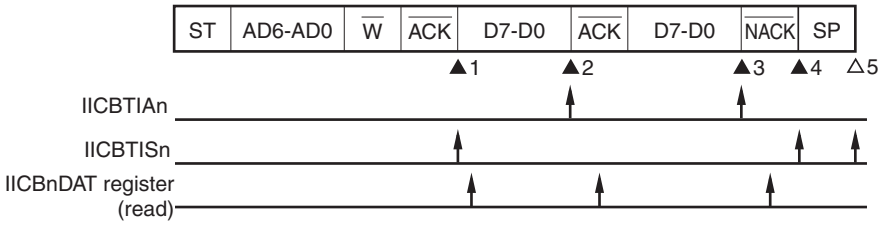
Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

26.8.10 Continuous transfer mode (slave device operation (reception): during slave address reception (IICBnSTR0.IICBnSSC0 bit = 1))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Address ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 0-100101 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100100 0100--00B]

IICBnDAT register read

→ IICBnSTR0 register = 0-000100 0100--00B

[▲3: IICBnSTR0 register = 0-100100 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 0-000100 0100-00B

▲4: IICBnSTR0 register = 0-010100 0100-00B

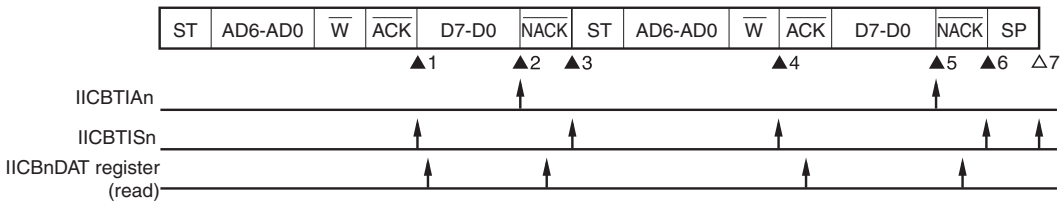
IICBnTRG.IICBnWRET bit = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, address match)



[▲1: IICBnSTR0 register = 0-110101 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100101 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

→ IICBnDAT register read

▲3: IICBnSTR0 register = 0-110101 0110--00B

IICBnTRG.IICBnWRET bit = 1

[▲4: IICBnSTR0 register = 0-100100 0110--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 0-000100 0110--00B

[▲5: IICBnSTR0 register = 0-100100 0100--00B]

▲6: IICBnSTR0 register = 0-010100 0100--00B

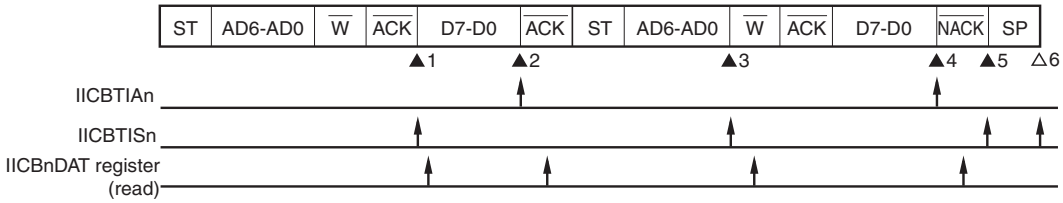
IICBnTRG.IICBnWRET bit = 1

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, extension code reception)



[▲1: IICBnSTR0 register = 0-100101 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100100 0100--00B]

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-100100 0110--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲4: IICBnSTR0 register = 0-100100 0110--00B]

IICBnDAT register read

▲5: IICBnSTR0 register = 0-111000 0100--00B

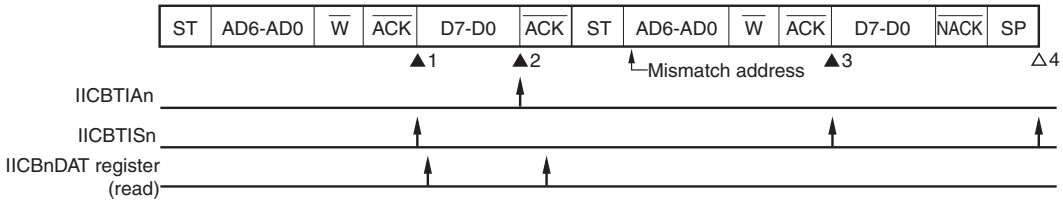
IICBnTRG.IICBnWRET bit = 1

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, address mismatch (extension code mismatch))



[▲1: IICBnSTR0 register = 0-000101 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100100 0100--00B]

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-000000 0110--00B]

△4: IICBnSTR0 register = 0-000000 0001--00B

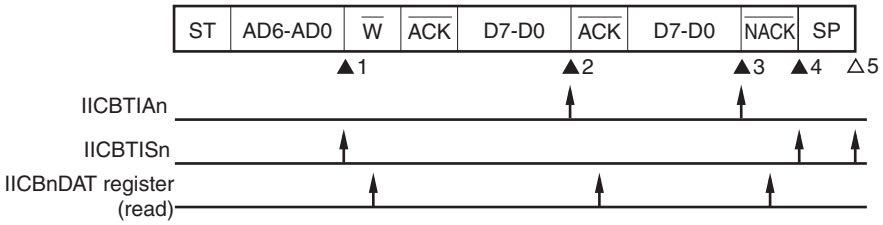
Note
- ▲ Always output
- △ Output only when IICBnCTL0.IICBnSLSI = 1
- \- Undefined
- X don't care

26.8.11 Continuous transfer mode (slave device operation (reception): during extension code reception (IICBnSTR0.IICBnSSEX bit = 1))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Code ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 0-101000 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-101001 0110--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-10001 0100--00B]

IICBnDAT register read

▲4: IICBnSTR0 register = 0-111000 0100--00B

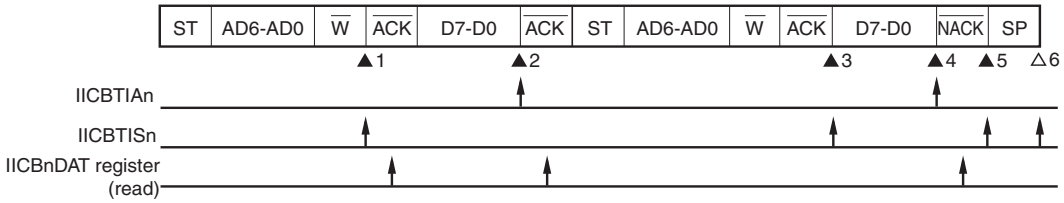
IICBnTRG.IICBnWRET bit = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, address match)



[▲1: IICBnSTR0 register = 0-101000 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-011000 0110--00B]

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-111001 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲4: IICBnSTR0 register = 0-010100 0110--00B]

IICBnDAT register read

▲5: IICBnSTR0 register = 0-110100 0100--00B

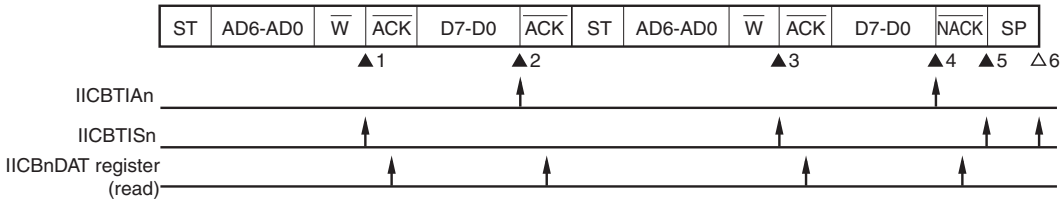
IICBnTRG.IICBnWRET bit = 1

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, extension code reception)



[▲1: IICBnSTR0 register = 0-101000 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-011001 0110--00B]

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-101000 0110--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲4: IICBnSTR0 register = 0-101001 0110--00B]

IICBnDAT register read

▲5: IICBnSTR0 register = 0-011000 0100--00B

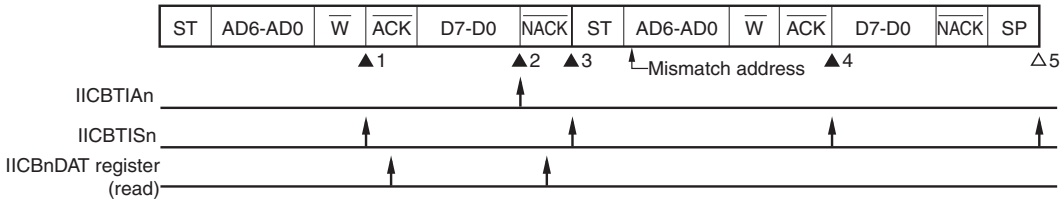
IICBnTRG.IICBnWRET bit = 1

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 0 (after restart, address mismatch (extension code mismatch))



[▲1: IICBnSTR0 register = 0-101000 0110--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-101001 0110--00B]

IICBnCTL0.IICBnSLAC bit = 0

▲3: IICBnSTR0 register = 0-010000 0100--00B

IICBnTRG.IICBnWRET bit = 1

[▲4: IICBnSTR0 register = 0-000000 0110--00B]

△5: IICBnSTR0 register = 0-000000 0001--00B

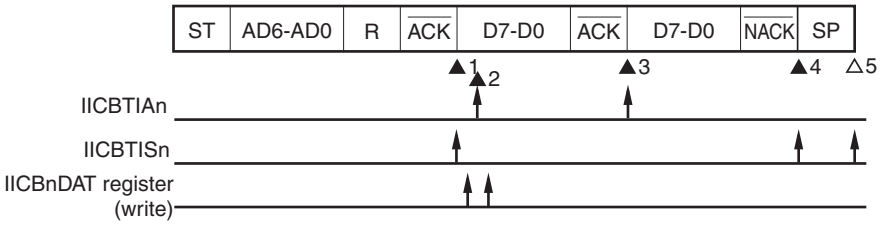
Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

26.8.12 Continuous transfer mode (slave device operation (transmission): during slave address reception (IICBnSTR0.IICBnSSC0 bit = 1))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Address ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-110111 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-00011X 0100--00B]

IICBnDAT register write

→ IICBnSTR0 register = 0-100011X 0100--00B

▲3: IICBnSTR0 register = 0-000111 0100--00B

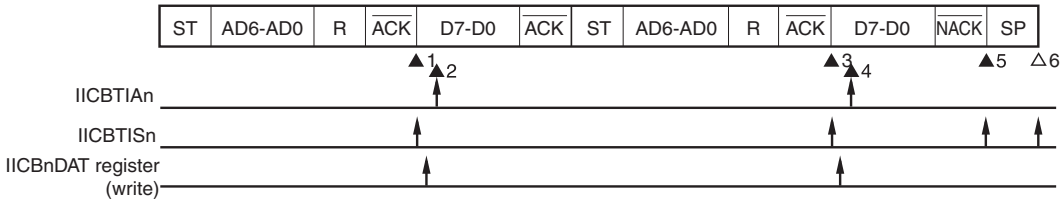
▲4: IICBnSTR0 register = 0-010110 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, address match)



▲1: IICBnSTR0 register = 0-010111 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-00111X 01X0--00B]

▲3: IICBnSTR0 register = 0-010111 0110--00B

IICBnDAT register write

[▲4: IICBnSTR0 register = 0-100101 01X0--00B]

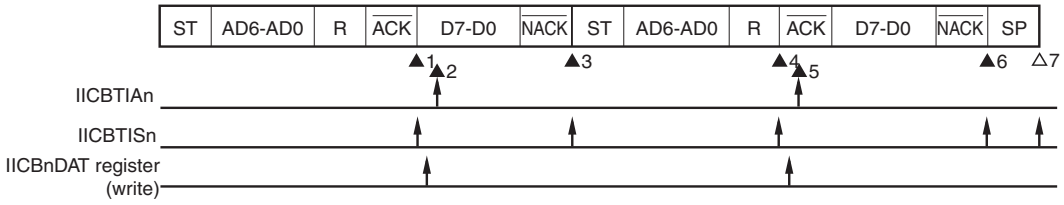
▲5: IICBnSTR0 register = 0-110100 0100--00B

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-110111 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-100111 0100--00B]

▲3: IICBnSTR0 register = 0-111010 0110--00B

▲4: IICBnSTR0 register = 0-111010 0110--00B

IICBnDAT register write

[▲5: IICBnSTR0 register = 0-111011 0110--00B]

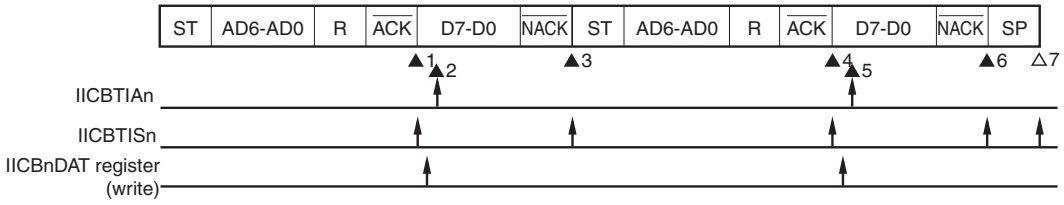
▲6: IICBnSTR0 register = 0-111010 0100--00B

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-110111 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-100111 0100--00B]

▲3: IICBnSTR0 register = 0-000010 0100--00B

▲4: IICBnSTR0 register = 0-000011 0110--00B

IICBnDAT register write

[▲5: IICBnSTR0 register = 0-00001X 0100--00B]

▲6: IICBnSTR0 register = 0-000010 0100--00B

△7: IICBnSTR0 register = 0-000000 0001--00B

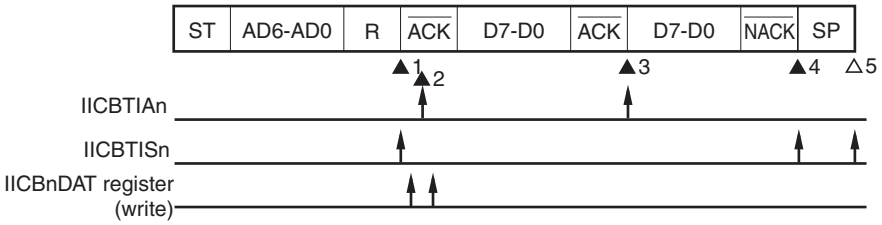
Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

26.8.13 Continuous transfer mode (slave device operation (transmission): during extension code reception (IICBnSTR0.IICBnSSEX bit = 1))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Code ~ Data ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-011010 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-011011 0110--00B]

IICBnDAT register write

[▲3: IICBnSTR0 register = 0-011011 0100--00B]

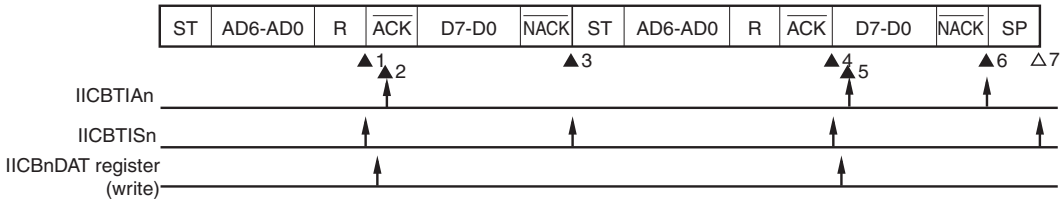
▲4: IICBnSTR0 register = 0-111010 0100--00B

△5: IICBnSTR0 register = 0-000010 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, address match)



▲1: IICBnSTR0 register = 0-011000 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-011001 0110--00B]

▲3: IICBnSTR0 register = 0-011000 0100--00B

▲4: IICBnSTR0 register = 0-010101 0110--00B

IICBnDAT register write

[▲5: IICBnSTR0 register = 0-010101 0110--00B]

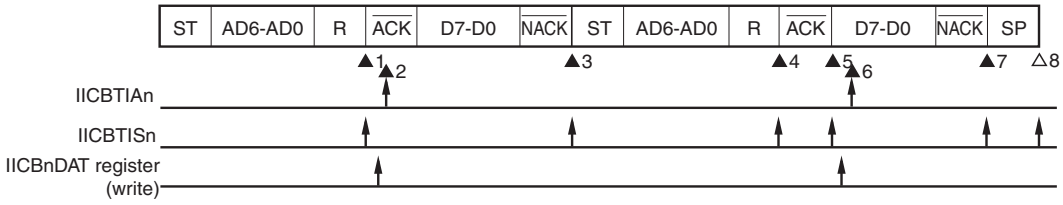
▲6: IICBnSTR0 register = 0-010100 0100--00B

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-011000 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-011001 0110--00B]

▲3: IICBnSTR0 register = 0-011000 0100--00B

▲4: IICBnSTR0 register = 0-011000 0110--00B

▲5: IICBnSTR0 register = 0-011001 0110--00B

IICBnDAT register write

[▲6: IICBnSTR0 register = 0-011001 0110--00B]

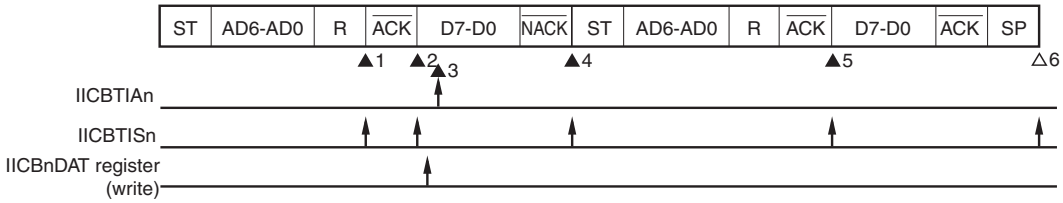
▲7: IICBnSTR0 register = 0-011000 0100--00B

△8: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

(a) When IICBnCTL0.IICBnSLWT bit is 1 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-011000 0110--00B

▲2: IICBnSTR0 register = 0-011001 0110--00B

IICBnDAT register write

[▲3: IICBnSTR0 register = 0-011010 0100--00B]

▲4: IICBnSTR0 register = 0-000000 0100--00B

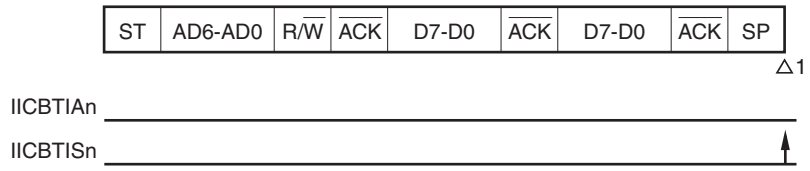
▲5: IICBnSTR0 register = 0-000000 0110--00B

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

26.8.14 Continuous transfer mode (non-participation in communications)

(1) Start ~ Code ~ Data ~ Data ~ Stop



△1: IICBnSTR0 register = 0-0000X0 0001--00B

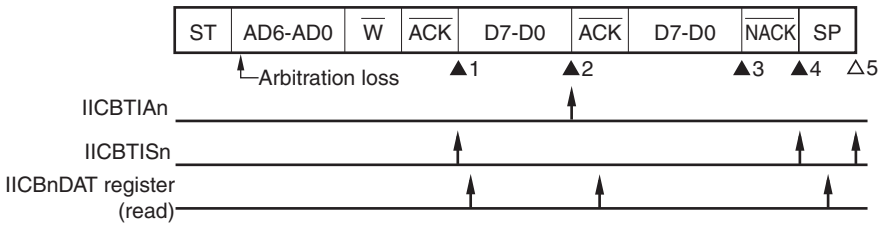
Note △ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

26.8.15 Continuous transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1) (when address was transferred during reception): operation as slave after arbitration loss)

When using IICBn as the master in a multi-master system, read the IICBnSTR0.IICBnALDF bit for each IICBTISn interrupt occurrence to confirm the arbitration result.

(1) Address match after arbitration loss

(a) During reception, when IICBnCTL0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 0-100101 0110--01B]

IICBnSTRC.IICBnCLAF bit = 1

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100101 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-100100 0100--00B]

IICBnDAT register read

▲4: IICBnSTR0 register = 0-010100 0100--00B

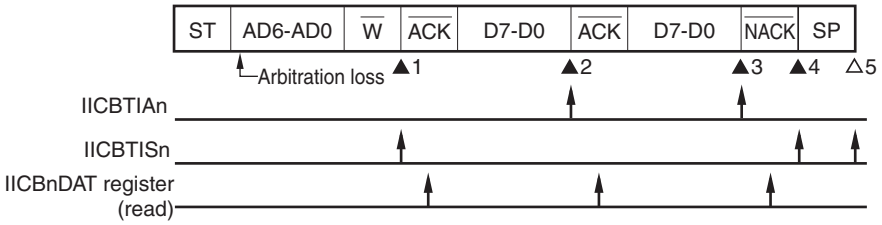
IICBnTRG.IICBnWRET bit = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(2) Upon extension code detection after arbitration loss

(a) During reception, when IICBnCTL0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 0-101000 0110--01B]

IICBnSTRC.IICBnCLAF bit = 1

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-101000 0110--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-101000 0100--00B]

IICBnDAT register read

▲4: IICBnSTR0 register = 0-011000 0100--00B]

IICBnTRG.IICBnWRET bit = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

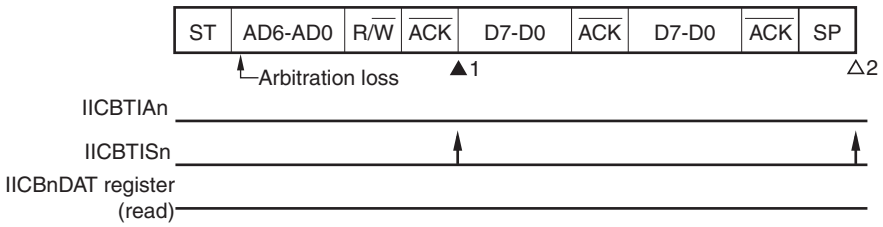
Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

26.8.16 Continuous transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1) (when address was transferred during reception): non-participation in communications after arbitration loss)

When using IICBn as the master in a multi-master system, read the IICBnSTR0.IICBnALDF bit for each IICBTISn interrupt occurrence to confirm the arbitration result.

(1) Arbitration loss during slave address transmission

(a) During reception, when IICBnCTL0.IICBnSLWT bit is 0



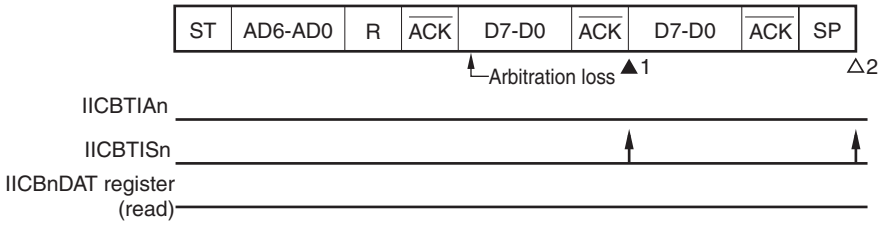
▲1: IICBnSTR0 register = 0-000001 0110--01B (IICBnSTRC.IICBnCLAF bit = 1)

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(2) Arbitration loss during data transfer

(a) During reception, when IICBnCTL0.IICBnSLWT bit is 1



[▲1: IICBnSTR0 register = 0-000000 0100--01B]

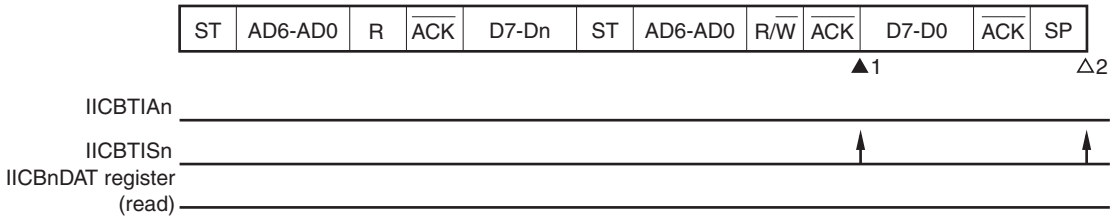
IICBnSTRC.IICBnCLAF bit = 1

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(3) Arbitration loss for the restart condition during data transfer

(a) During reception, when IICBnCTL0.IICBnSLWT bit is 1 (extension code mismatch, address mismatch)



[▲1: IICBnSTR0 register = 0-000001 0100--01B]

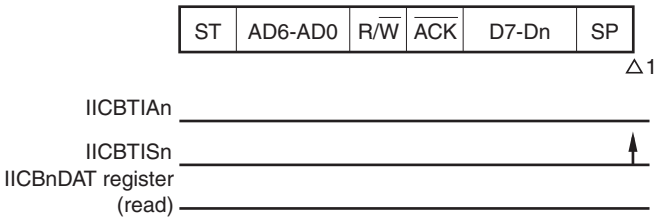
IICBnSTRC.IICBnCLAF bit = 1

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined

(4) Arbitration loss for the stop condition during data transfer

(a) During reception, when IICBnCTL0.IICBnSLWT bit is 1



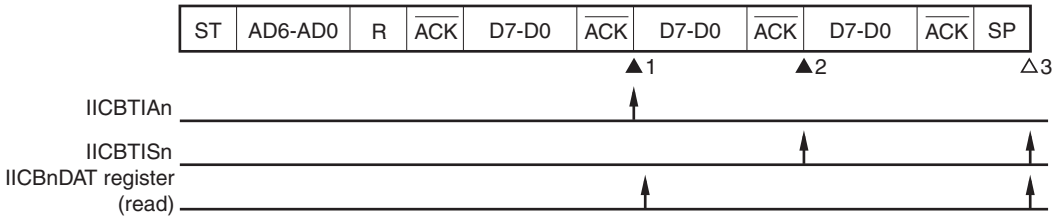
△1: IICBnSTR0 register = 0-000000 0001--01B

IICBnSTRC.IICBnCLAF bit = 1

Note △ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined

(5) Arbitration loss because the SDAn signal is low level when attempting to output restart condition

(a) When IICBnCTL0.IICBnSLWT bit is 1



[▲1: IICBnSTR0 register = 1-1000XX 0100--00B]

IICBnDAT register read

IICBnTRG.IICBnSTT bit = 1

▲2: IICBnSTR0 register = 0-000000 0100--01B

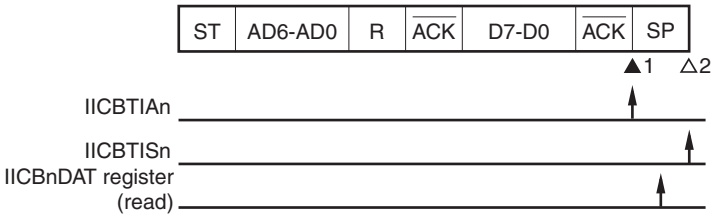
IICBnSTRC.IICBnCLAF bit = 1

△3: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(6) Arbitration loss for the stop condition when attempting to output restart condition

(a) When IICBnCTL0.IICBnSLWT bit is 1



[▲1: IICBnSTR0 register = 1-000001 0100--00B]

IICBnDAT register read

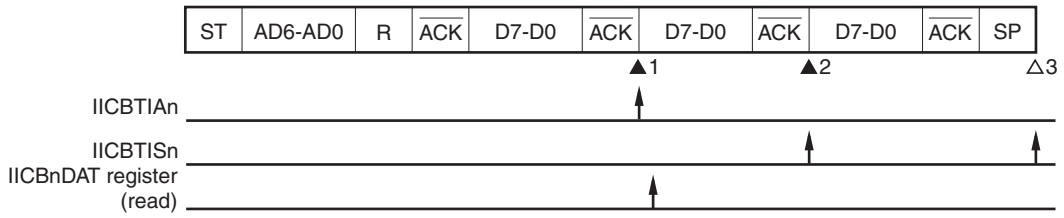
IICBnTRG.IICBnSTT bit = 1

△2: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined

(7) Arbitration loss because the SDAn signal is low level when attempting to output stop condition

(a) When IICBnCTL0.IICBnSLWT bit is 1



[▲1: IICBnSTR0 register = 1-1000XX 0100--00B]

IICBnDAT register read

IICBnTRG.IICBnSPT bit = 1

[▲2: IICBnSTR0 register = 0-0000XX 0100--01B (IICBnSTRC.IICBnCLAF bit = 1)

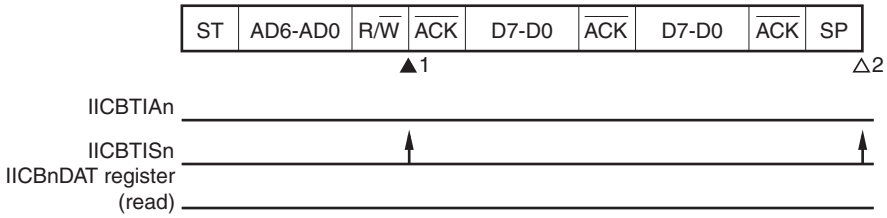
△3: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don't care

26.8.17 Continuous transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1) (when address was transferred during reception): non-participation in communications after arbitration loss (during extension code transfer))

When using IICBn as the master in a multi-master system, read the IICBnSTR0.IICBnALDF bit for each IICBTISn interrupt occurrence to confirm the arbitration result.

(1) Arbitration loss during extension code transfer



[▲1: IICBnSTR0 register = 0-1000X0 0110--01B]

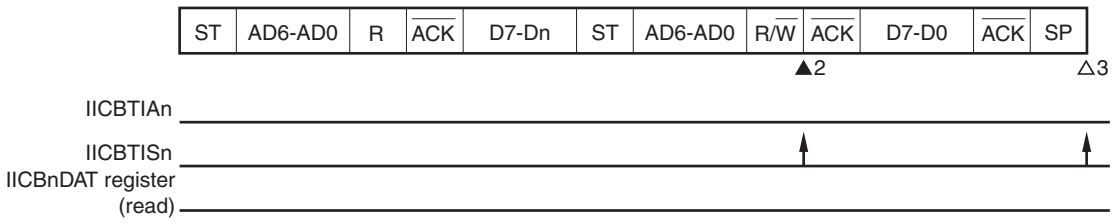
IICBnSTRC.IICBnCLAF bit = 1

IICBnTRG.IICBnLRET bit = 1

△2: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don’t care

(2) Arbitration loss for the restart condition during data transfer (extension code match)



▲1: IICBnSTR0 register = 1-0000X1 0110--00B

▲2: IICBnSTR0 register = 0-0100X0 0100--01B (IICBnSTRC.IICBnCLAF bit = 1, IICBnTRG.IICBnLRET bit = 1)

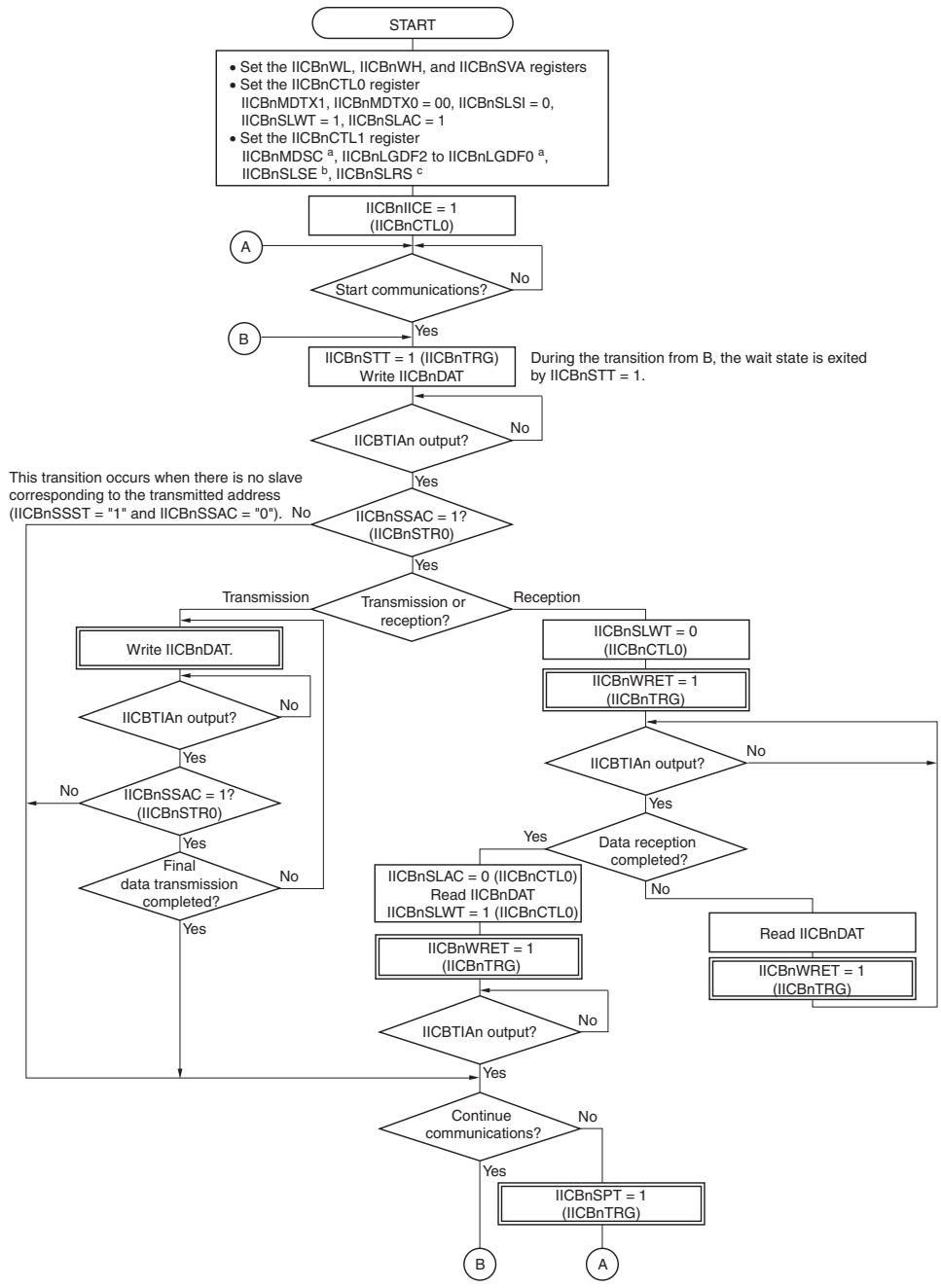
△3: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

26.9 Setting Sequence

26.9.1 Single master environment

(1) Master operate setting sequence during single transfer mode

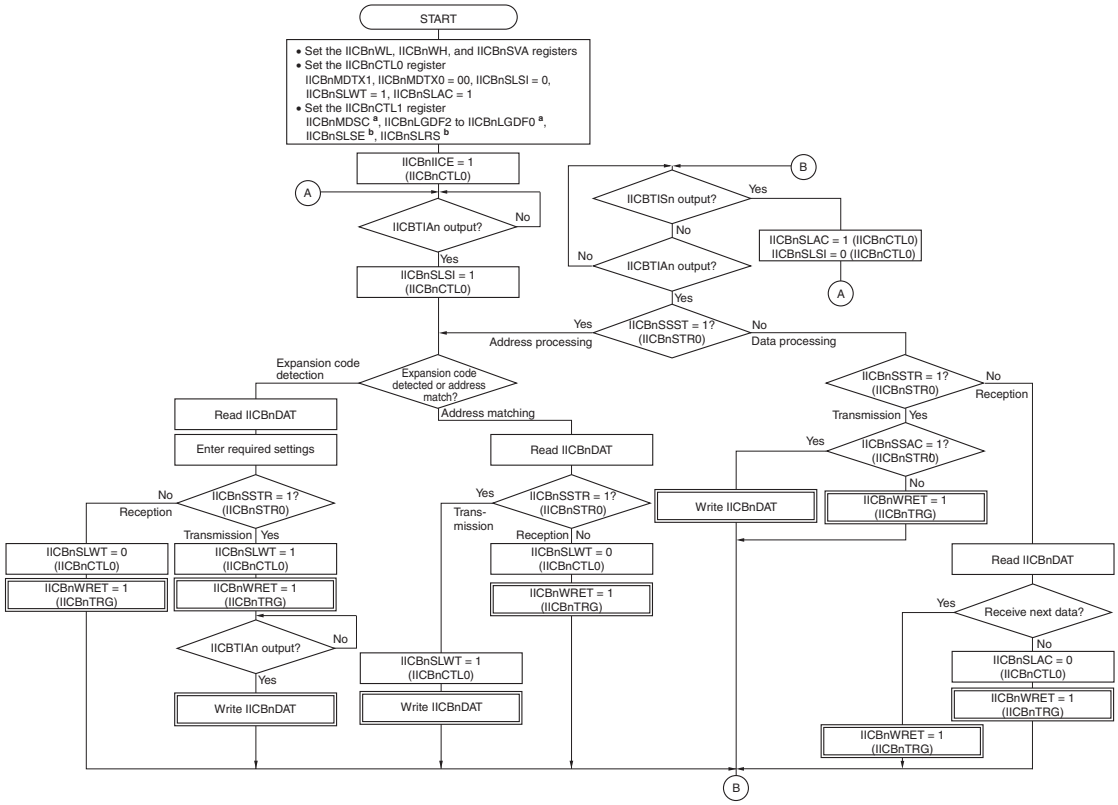


a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.
c) Values are not referenced in this environment.

Note The double-boxed items indicate processing that exits the wait state.

Figure 26-14 Master operate setting sequence during single transfer mode (single master environment)

(2) Slave operate setting sequence during single transfer mode



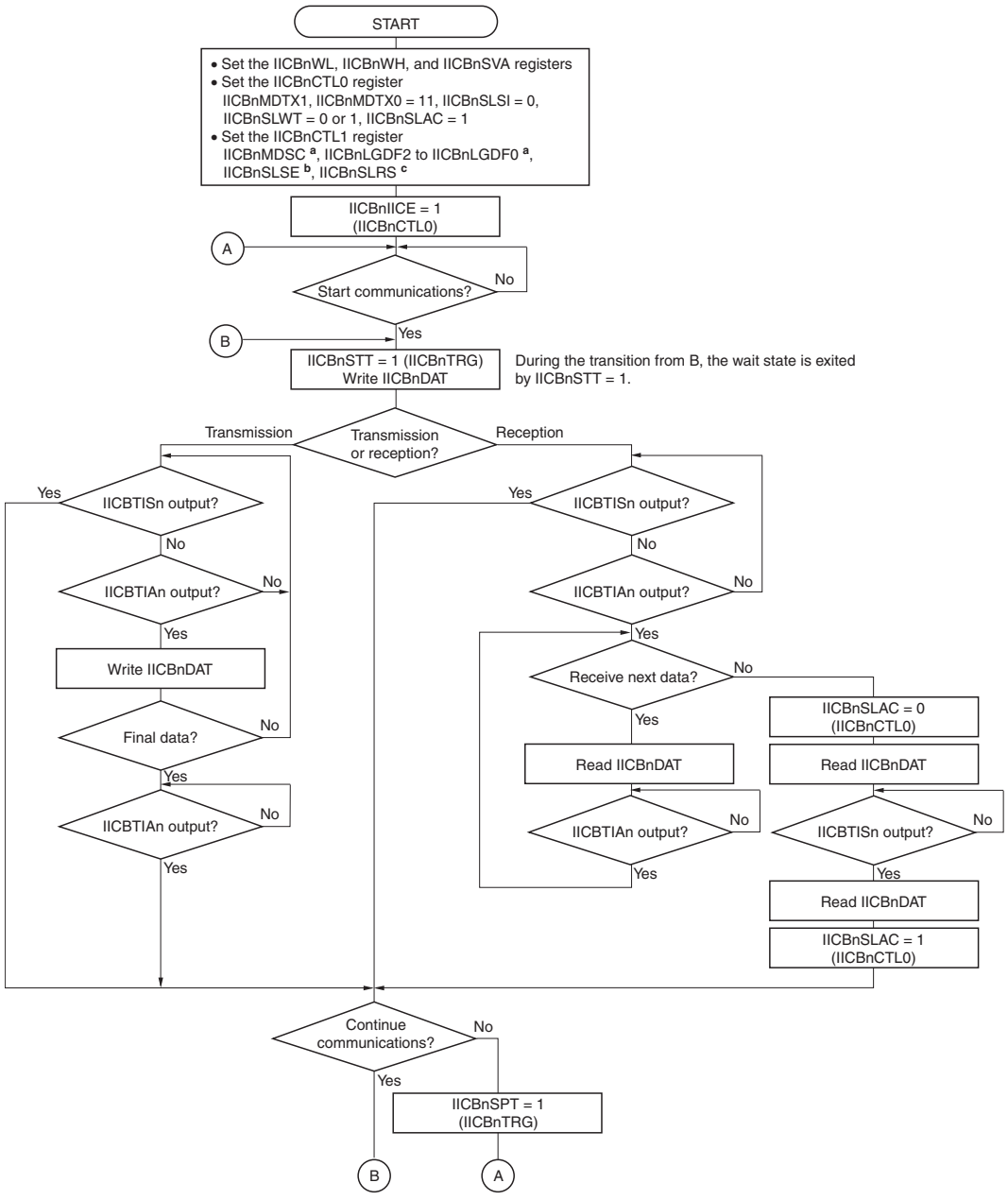
a) Enter settings according to the environment.
b) Values are not referenced in this environment.

Note The double-boxed items indicate processing that exits the wait state.

Figure 26-15 Slave operate setting sequence during single transfer mode (single master environment)

(3) Master operate setting sequence during continuous transfer mode

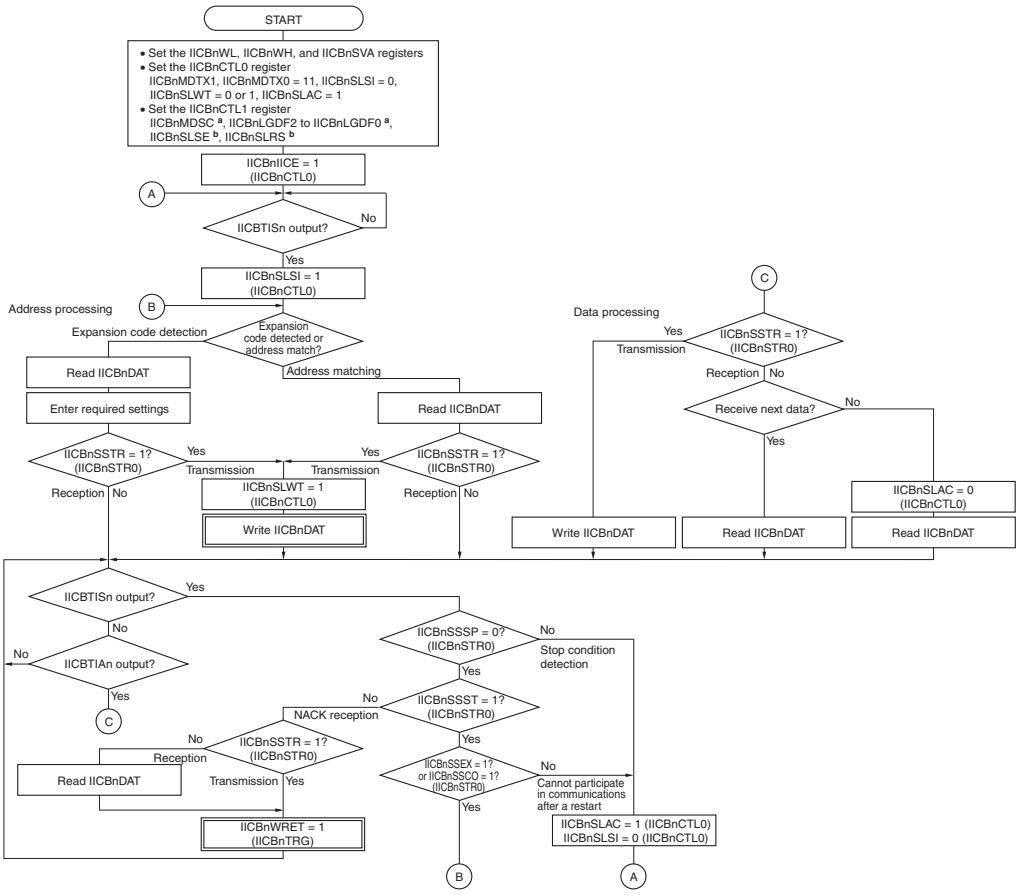
<R>



a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.
c) Values are not referenced in this environment.

Figure 26-16 Master operate setting sequence during continuous transfer mode (single master environment)

(4) Slave operate setting sequence during continuous transfer mode



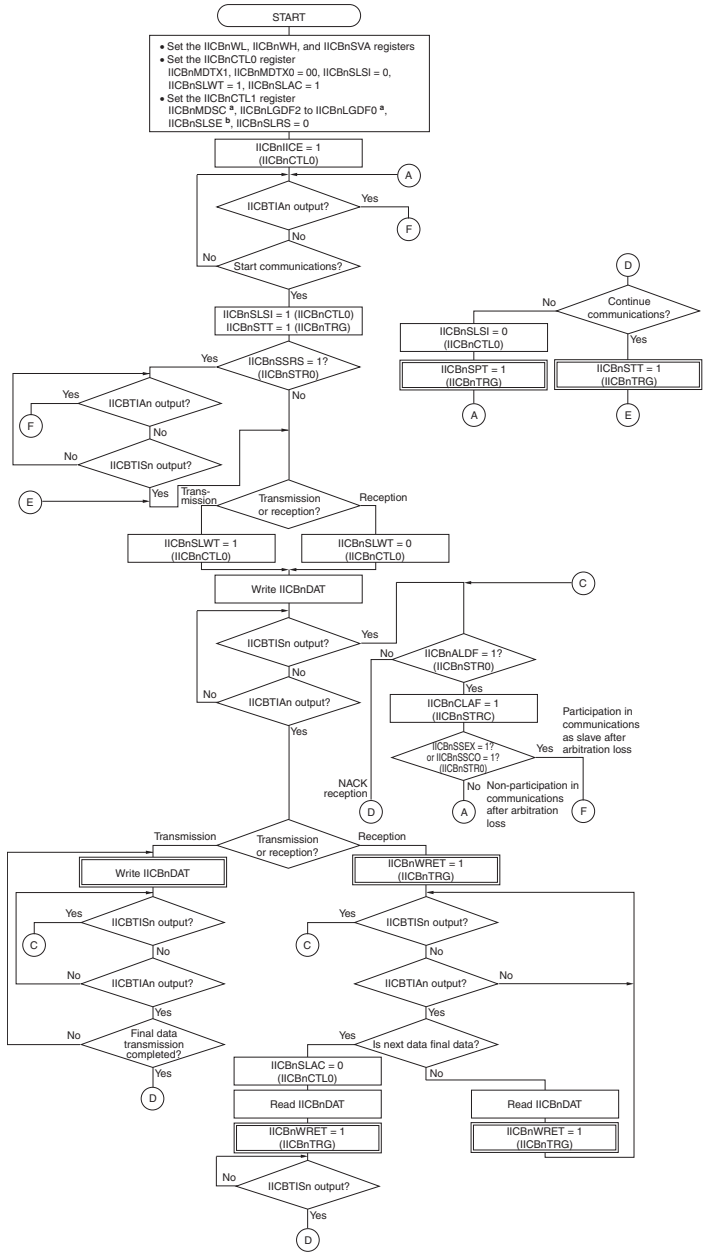
a) Enter settings according to the environment.
b) Values are not referenced in this environment.

Note The double-boxed items indicate processing that exits the wait state.

Figure 26-17 Slave operate setting sequence during continuous transfer mode (single master environment)

26.9.2 Multi-master environment

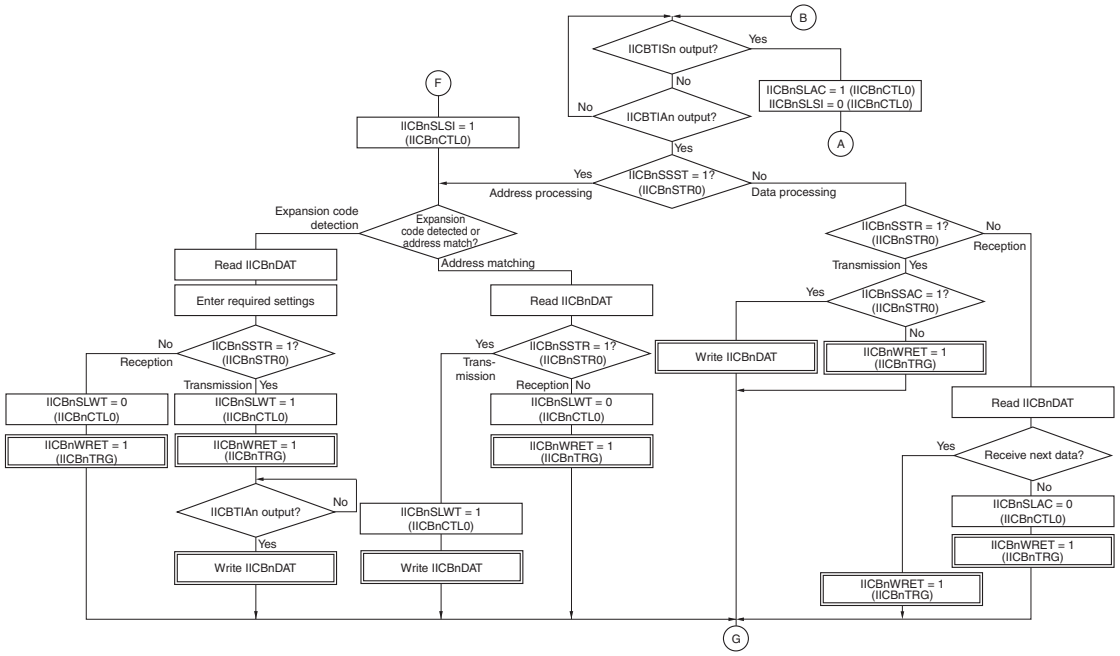
(1) Single transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0)



a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.

Note The double-boxed items indicate processing that exits the wait state.

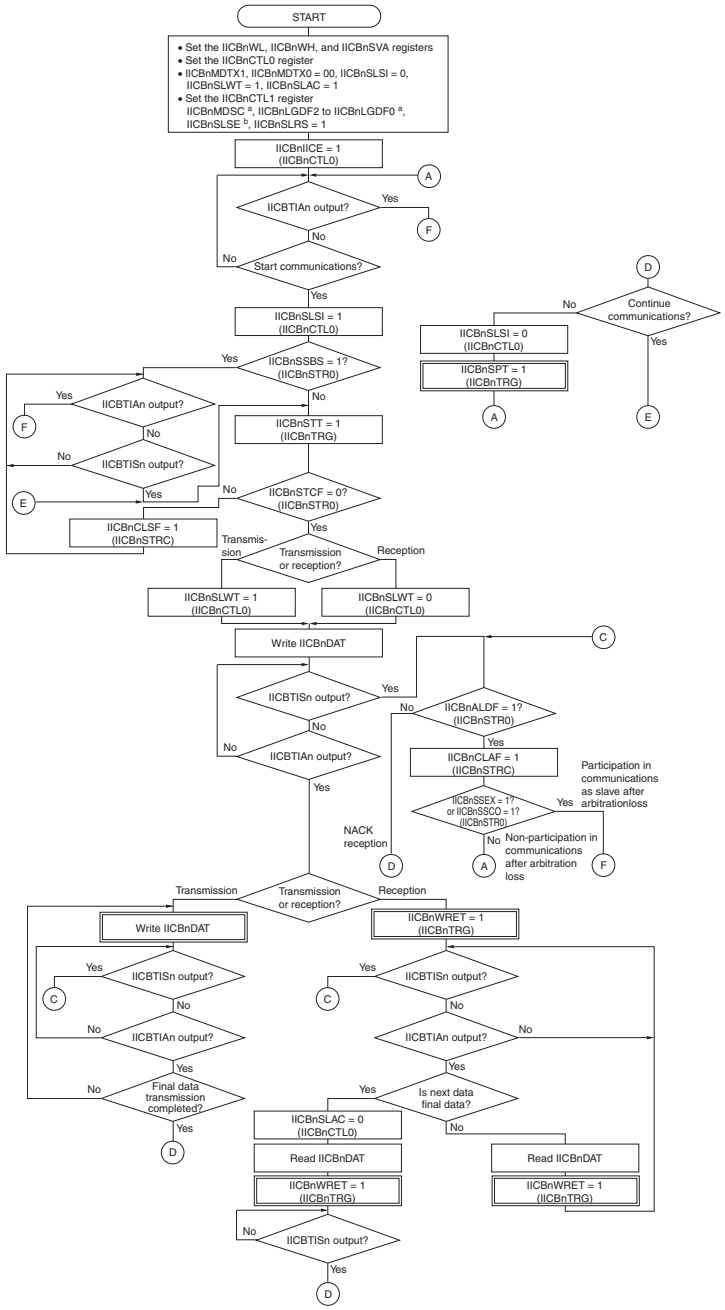
Figure 26-18 Single transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0) (multi-master environment) (1/2)



Note The double-boxed items indicate processing that exiting the wait state.

Figure 26-18 Single transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0) (multi-master environment) (2/2)

(2) Single transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1)

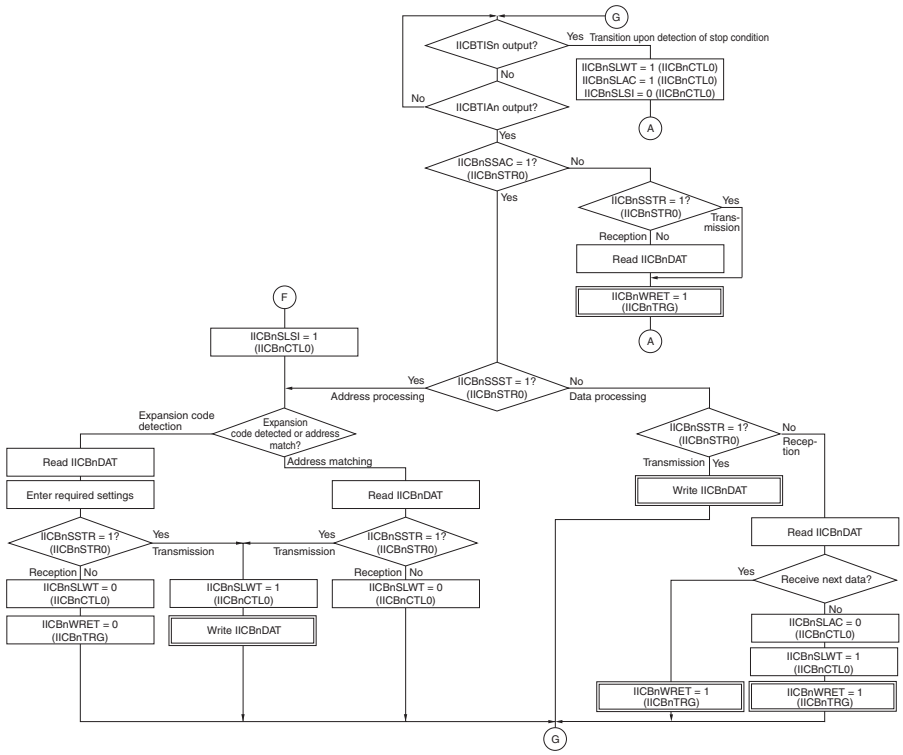


a) Enter settings according to the environment.

b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.

Note The double-boxed items indicate processing that exits the wait state.

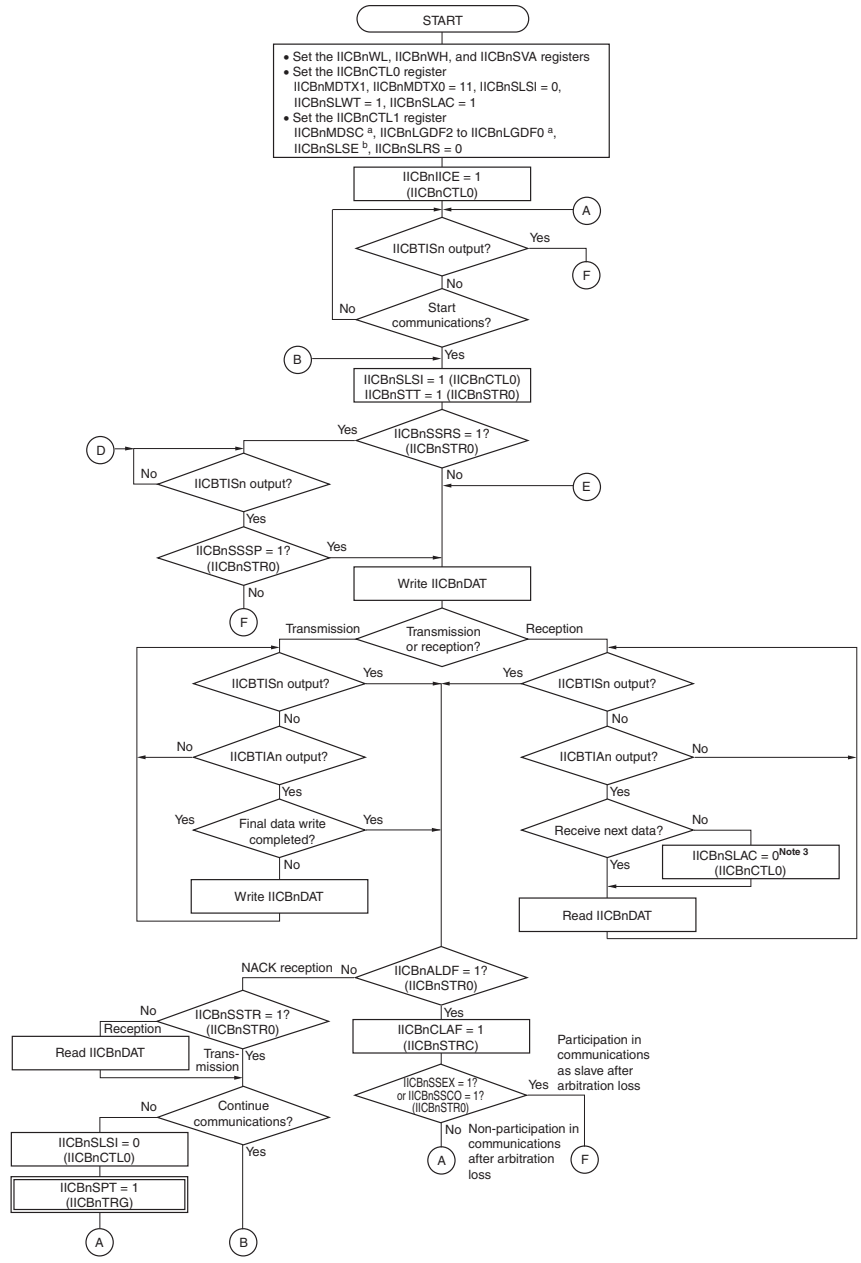
Figure 26-19 Single transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1) (multi-master environment) (1/2)



Note The double-boxed items indicate processing that exits the wait state.

Figure 26-19 Single transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1) (multi-master environment) (2/2)

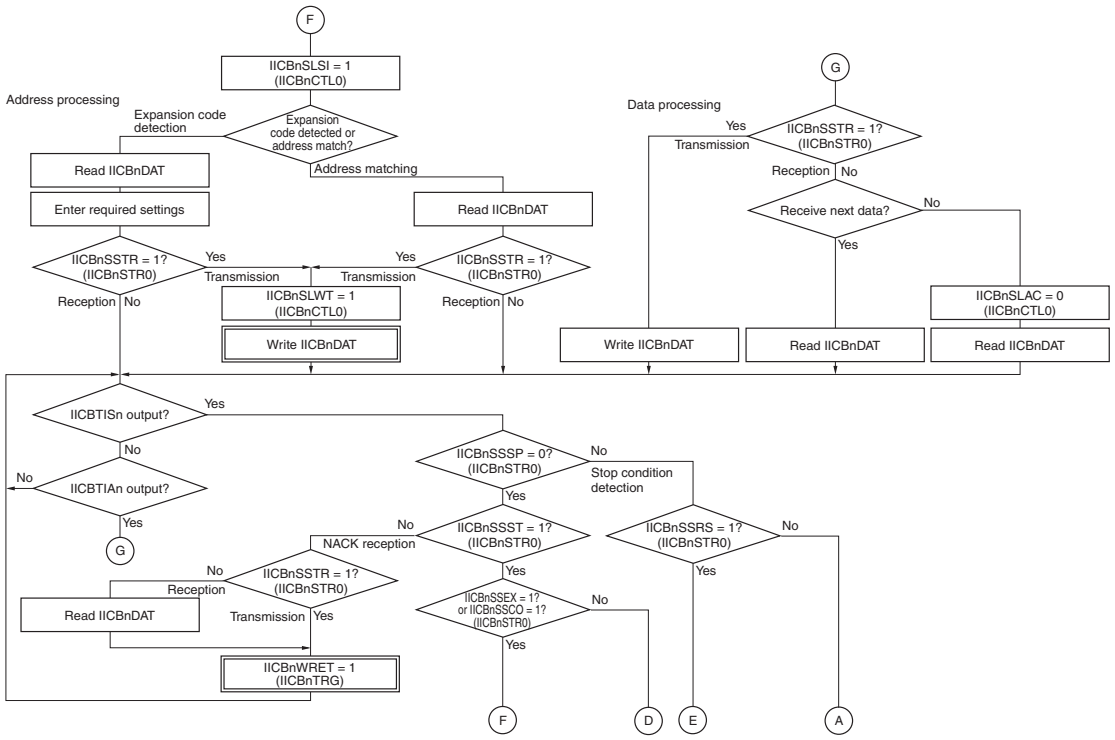
(3) Continuous transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0)



a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.
c) The IICBnCTL0.IICBnSLAC bit must be cleared to 0 before the 8th SCLn clock.

Note The double-boxed items indicate processing that exits the wait state.

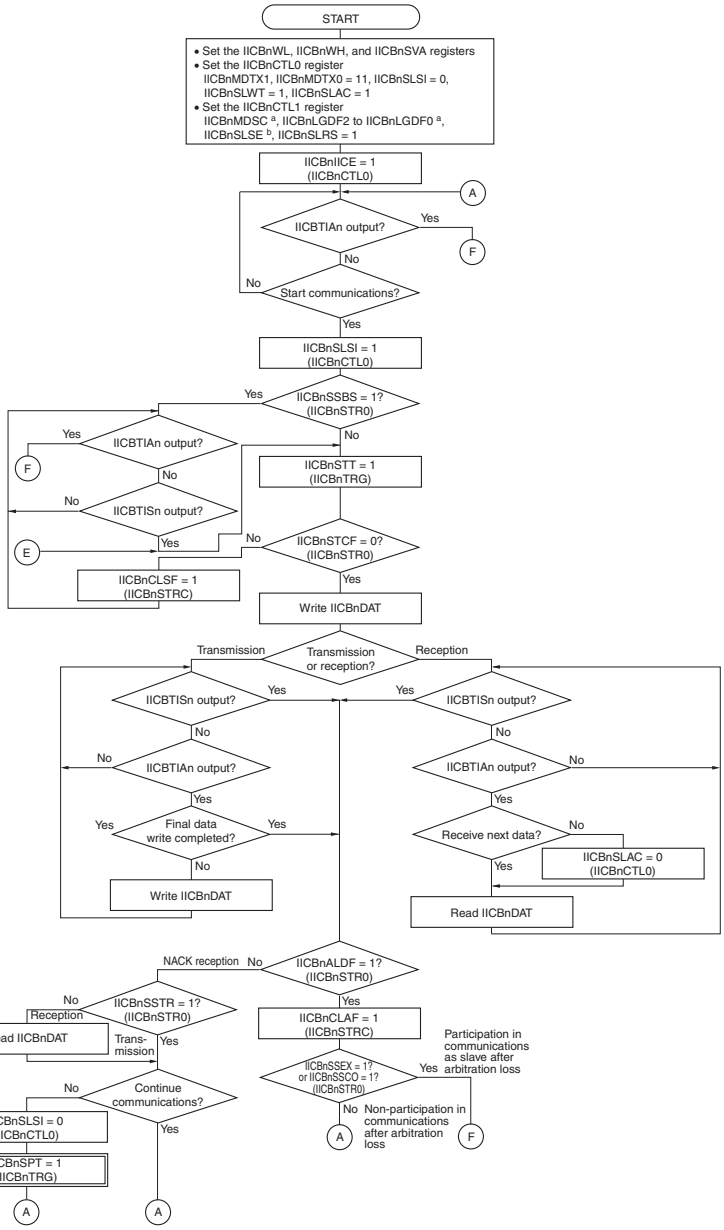
Figure 26-20 Continuous transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0) (multi-master environment) (1/2)



Note The double-boxed items indicate processing that exits the wait state.

Figure 26-20 Continuous transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0) (multi-master environment) (2/2)

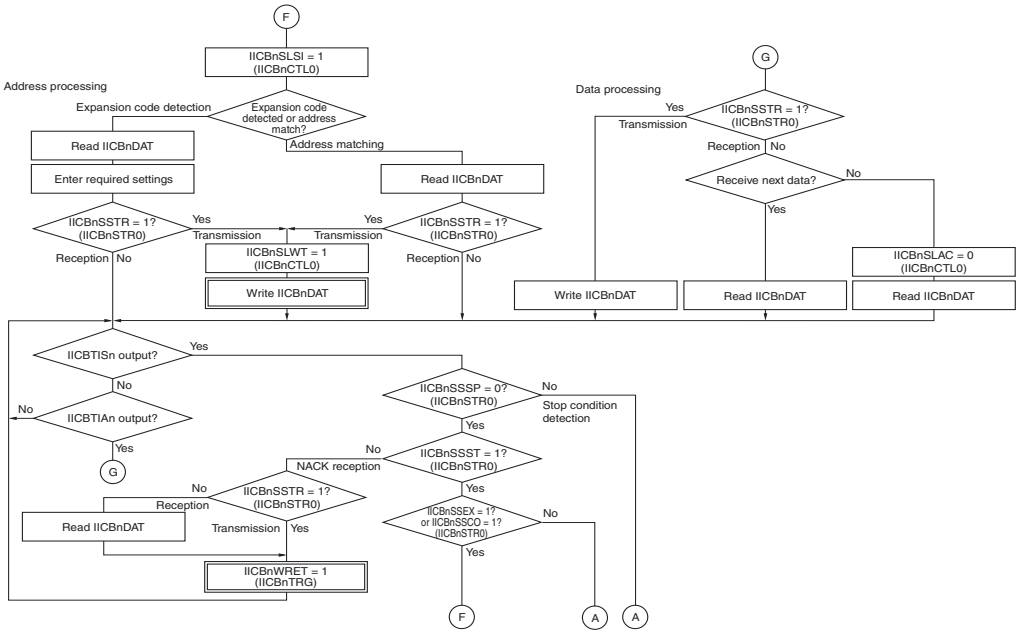
(4) Continuous transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1)



a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.

Note The double-boxed items indicate processing that exits the wait state.

Figure 26-21 Continuous transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1) (multi-master environment) (1/2)



Note The double-boxed items indicate processing that exits the wait state.

Figure 26-21 Continuous transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1) (multi-master environment) (2/2)

Chapter 27 IISA Interface (IISA)

This chapter describes the IISA interface (IISA).

The first section describes all properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names. The subsequent sections describe the features that apply to all implementations.

27.1 V850E2/Sx4-H IISA Features

Instances

This microcontroller has following number of instances of IISA:

Table 27-1 Instances of IISA

IISA	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of instances	4	4	6
Names	IISA0 to IISA3	IISA0 to IISA3	IISA0 to IISA5

Instances index n

Throughout this chapter, the individual instances of IISA are identified by the index “n” (n = 0 to 5), for example, IISAnCTL for the IISAn control register.

Register addresses

All IISAn register addresses are given as addresses offset from the individual base address <IISAn_base>.
The base address <IISAn_base> of each IISAn is listed in the following table:

Table 27-2 Register base addresses <IISAn_base>

IISAn	<IISAn_base> address
IISA0	FF82 8000_H
IISA1	FF82 8100_H
IISA2	FF82 8200_H
IISA3	FF82 8300_H
IISA4	FF82 8400_H
IISA5	FF82 8500_H

Clock supply The following clocks are supplied to IISA:

Table 27-3 IISAn clock supply

IISAn	Clock	Connected to:
IISA0	IISA0SCK	IISA serial clock selector[a]
	PCLK	Clock generator CKSCLK_102
IISA1	IISA1SCK	IISA serial clock selector
	PCLK	Clock generator CKSCLK_103
IISA2	IISA2SCK	IISA serial clock selector
	PCLK	Clock generator CKSCLK_105
IISA3	IISA3SCK	IISA serial clock selector
	PCLK	Clock generator CKSCLK_109
IISA4	IISA4SCK	IISA serial clock selector
	PCLK	Clock generator CKSCLK_117
IISA5	IISA5SCK	IISA serial clock selector
	PCLK	Clock generator CKSCLK_129

a) For details, see *27.1.2 "IISA master clock generator"*.

Interrupts and DMA IISA can generate following interrupt and DMA requests:

Table 27-4 IISA interrupt and DMA requests (1/2)

IISAn signals	Function	Connected to:
IISA0:		
IISA0TIA	General interrupt	Interrupt controller INTIISA0IA
IISA0TITXU	Transmission FIFO underrun interrupt	Interrupt controller INTIISA0ITXU
IISA0TITXT	Transmission FIFO empty trigger level interrupt	Interrupt controller INTIISA0ITXT DMA controller trigger 28
IISA0TIRXO	Reception FIFO overrun interrupt	Interrupt controller INTIISA0IRXO
IISA0TIRXT	Reception FIFO full trigger level interrupt	Interrupt controller INTIISA0IRXT DMA controller trigger 29
IISA0TIFERR	Framing error interrupt	Interrupt controller INTIISA0IFERR
IISA1:		
IISA1TIA	General interrupt	Interrupt controller INTIISA1IA
IISA1TITXU	Transmission FIFO underrun interrupt	Interrupt controller INTIISA1ITXU
IISA1TITXT	Transmission FIFO empty trigger level interrupt	Interrupt controller INTIISA1ITXT DMA controller trigger 30
IISA1TIRXO	Reception FIFO overrun interrupt	Interrupt controller INTIISA1IRXO
IISA1TIRXT	Reception FIFO full trigger level interrupt	Interrupt controller INTIISA1IRXT DMA controller trigger 31
IISA1TIFERR	Framing error interrupt	Interrupt controller INTIISA1IFERR

Table 27-4 IISA interrupt and DMA requests (2/2)

IISAn signals	Function	Connected to:
IISA2:		
IISA2TIA	General interrupt	Interrupt controller INTIISA2IA
IISA2TITXU	Transmission FIFO underrun interrupt	Interrupt controller INTIISA2ITXU
IISA2TITXT	Transmission FIFO empty trigger level interrupt	Interrupt controller INTIISA2ITXT DMA controller trigger 32
IISA2TIRXO	Reception FIFO overrun interrupt	Interrupt controller INTIISA2IRXO
IISA2TIRXT	Reception FIFO full trigger level interrupt	Interrupt controller INTIISA2IRXT DMA controller trigger 33
IISA2TIFERR	Framing error interrupt	Interrupt controller INTIISA2IFERR
IISA3:		
IISA3TIA	General interrupt	Interrupt controller INTIISA3IA
IISA3TITXU	Transmission FIFO underrun interrupt	Interrupt controller INTIISA3ITXU
IISA3TITXT	Transmission FIFO empty trigger level interrupt	Interrupt controller INTIISA3ITXT DMA controller trigger 34
IISA3TIRXO	Reception FIFO overrun interrupt	Interrupt controller INTIISA3IRXO
IISA3TIRXT	Reception FIFO full trigger level interrupt	Interrupt controller INTIISA3IRXT DMA controller trigger 35
IISA3TIFERR	Framing error interrupt	Interrupt controller INTIISA3IFERR
IISA4:		
IISA4TIA	General interrupt	Interrupt controller INTIISA4IA
IISA4TITXU	Transmission FIFO underrun interrupt	Interrupt controller INTIISA4ITXU
IISA4TITXT	Transmission FIFO empty trigger level interrupt	Interrupt controller INTIISA4ITXT DMA controller trigger 36
IISA4TIRXO	Reception FIFO overrun interrupt	Interrupt controller INTIISA4IRXO
IISA4TIRXT	Reception FIFO full trigger level interrupt	Interrupt controller INTIISA4IRXT DMA controller trigger 37
IISA4TIFERR	Framing error interrupt	Interrupt controller INTIISA4IFERR
IISA5:		
IISA5TIA	General interrupt	Interrupt controller INTIISA5IA
IISA5TITXU	Transmission FIFO underrun interrupt	Interrupt controller INTIISA5ITXU
IISA5TITXT	Transmission FIFO empty trigger level interrupt	Interrupt controller INTIISA5ITXT DMA controller trigger 64
IISA5TIRXO	Reception FIFO overrun interrupt	Interrupt controller INTIISA5IRXO
IISA5TIRXT	Reception FIFO full trigger level interrupt	Interrupt controller INTIISA5IRXT DMA controller trigger 64
IISA5TIFERR	Framing error interrupt	Interrupt controller INTIISA5IFERR

IISA hardware reset

IISA and its registers are initialized by the following reset signal:

Table 27-5 IISAn reset signal

IISAn	Reset signal
IISAn	System reset SYSRES

I/O signals

The I/O signals of IISA are listed in the following table:

Table 27-6 IISA I/O signals (1/2)

IISAn signals	Function	Connected to:
Common to IISAn:		
IISAACKI	Audio system clock I/O	Port IISAACK
IISAACKO		
IISA0:		
IISA0SCKI	Serial clock I/O	Port IISA0SCK
IISA0SCKO		
IISA0WS	Word select signal I/O	Port IISA0WS
IISA0STDI	Serial data input	Port IISA0SDI
IISA0STDO	Serial data output	Port IISA0SDO
IISA1:		
IISA1SCKI	Serial clock I/O	Port IISA1SCK
IISA1SCKO		
IISA1WS	Word select signal I/O	Port IISA1WS
IISA1STDI	Serial data input	Port IISA1SDI
IISA1STDO	Serial data output	Port IISA1SDO
IISA2:		
IISA2SCKI	Serial clock I/O	Port IISA2SCK
IISA2SCKO		
IISA2WS	Word select signal I/O	Port IISA2WS
IISA2STDI	Serial data input	Port IISA2SDI
IISA2STDO	Serial data output	Port IISA2SDO
IISA3:		
IISA3SCKI	Serial clock I/O	Port IISA3SCK
IISA3SCKO		
IISA3WS	Word select signal I/O	Port IISA3WS
IISA3STDI	Serial data input	Port IISA3SDI
IISA3STDO	Serial data output	Port IISA0SDO
IISA4:		
IISA4SCKI	Serial clock I/O	Port IISA4SCK
IISA4SCKO		
IISA4WS	Word select signal I/O	Port IISA4WS
IISA4STDI	Serial data input	Port IISA4SDI
IISA4STDO	Serial data output	Port IISA4SDO

Table 27-6 IISA I/O signals (2/2)

IISAn signals	Function	Connected to:
IISA5:		
IISA5SCKI	Serial clock I/O	Port IISA5SCK
IISA5SCKO		
IISA5WS	Word select signal I/O	Port IISA5WS
IISA5STDI	Serial data input	Port IISA5SDI
IISA3STDO	Serial data output	Port IISA5SDO

Clock frequency measurement

The IISAn serial clock input signals are internally connected to the capture input pins of timer array unit A.

Table 27-7 IISAn interrupt requests and DMA requests

IISAn signals	Function	Connected to:
IISA0:		
IISA0SCKI	Serial clock input	TAUA0 TAUA0TTIN2
IISA1:		
IISA1SCKI	Serial clock input	TAUA0 TAUA0TTIN3

Note For details, see *15.2 “TAUA Input Selection”*.

27.1.1 Serial clock selectors

IISA uses the serial clock IISAnSCK as the reference clock for all reception and transmission operations.

IISA can be operated in master and slave mode allowing different clock sources to be used. The clock selector provides the correct clocks to be used.

(1) Serial clock signals in master mode

In master mode (IISAnCTL.IISAnMD = 1), IISA can internally generate the audio clock IISAACK and the serial clock IISAnSCK by selecting and dividing an external or internal clock source signal.

- IISAACK can be output as IISAACKO to provide the audio clock for devices without their own clock generators.
- IISAnSCK is output as IISAnSCKO allowing slaves to synchronize with IISA.

Sampling frequency

The sampling frequency (Fs) is the frequency of the word select signal IISAnWS. The sampling frequency depends on the audio clock and the serial clock.

Typically, the clock source signal must be scaled down to achieve the required sampling frequency (Fs), audio clock IISAACK (= a x Fs), and serial clock IISAnSCK (= s x Fs).

Selecting and downscaling the clock source signal is done by the IISA master clock generator.

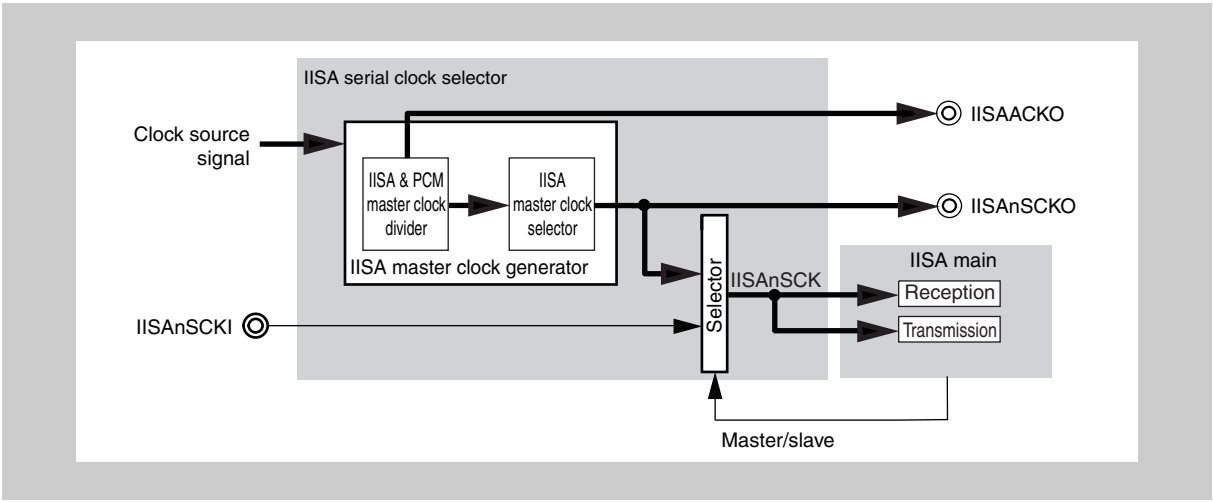


Figure 27-1 Serial clock signals in master mode

(2) Serial clock signals in slave mode

In slave mode (IISAnCTL.IISAnMD = 0), IISA generates the serial clock IISAnSCK by using the clock input to the external device pin IISAnSCKI. The audio clock IISAACK is not used.

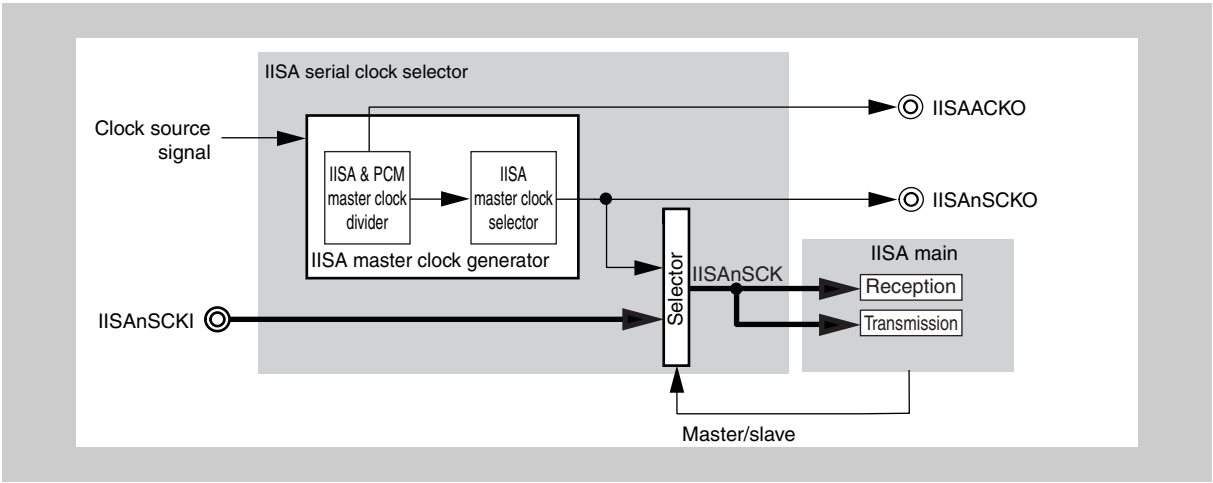


Figure 27-2 Serial clock signals in slave mode

27.1.2 IISA master clock generator

In master mode (IISAnCTL.IISAnMD = 1), IISA can internally generate the audio clock IISAACK and the serial clock IISAnSCK by selecting and dividing an external or internal clock source signal.

The IISA and PCM master clock dividers, which are included in the IISA master clock generator, can also be used to generate the PCM master clock.

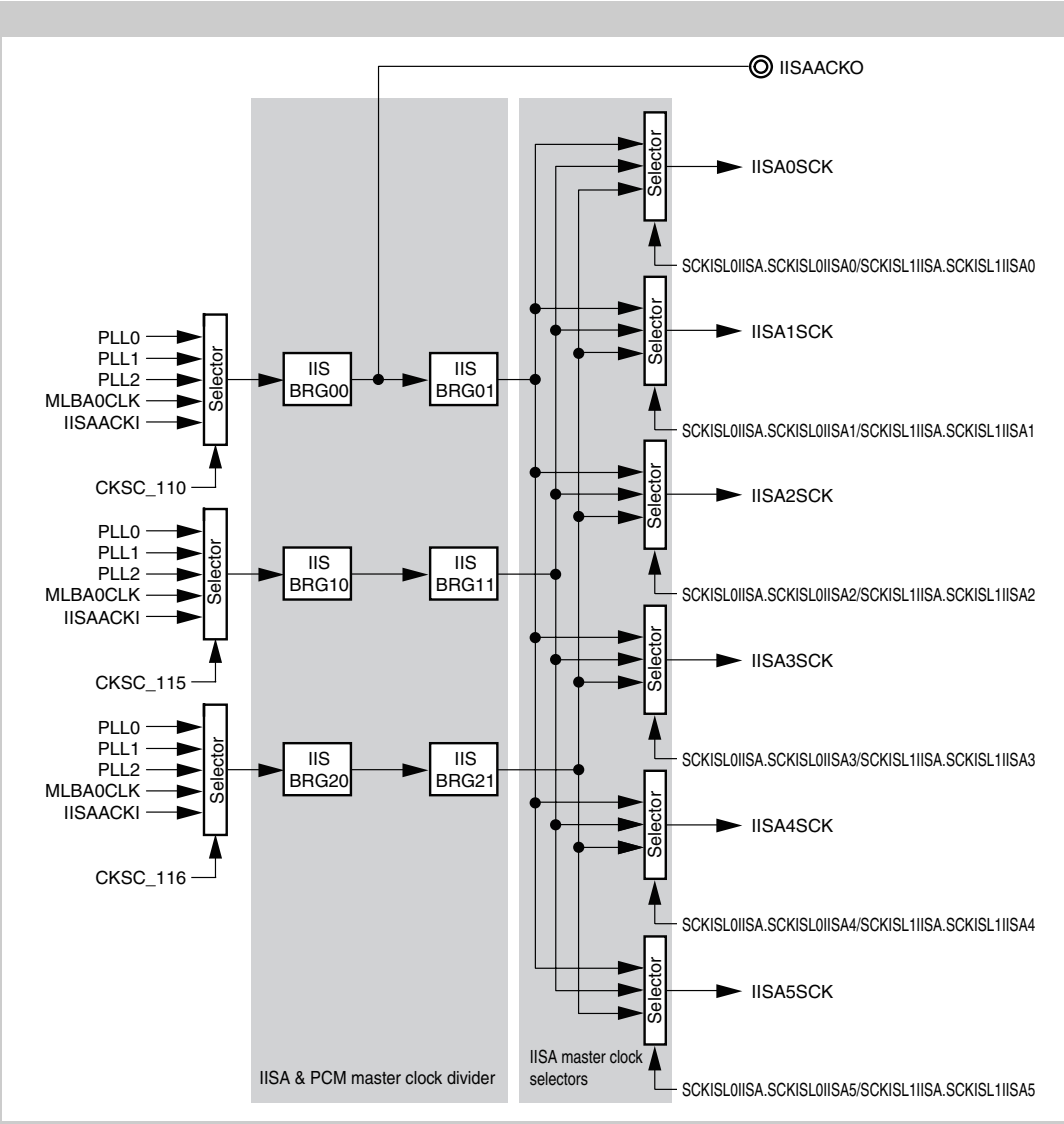


Figure 27-3 IISA master clock generator

(1) Source clock signals

The following singals can be selected as the clock sources:

- PLL0 to PLL2
- MLBA0CLK pin
- IISAACKI pin

Use the clock selection control registers CKSC_110, CKSC_115, and CKSC_116 to select the clock source signal. For details, see *9.4.3 “Clock domains of Isolated area 1”*.

(2) Baudrate generators

The V850E2/Sx4-H incorporates six baudrate generators IISBRGn (n = 00, 01, 10, 11, 20, 21). The baudrate generators divide the clock source signals to generate three serial clock signals.

The divisor value N is defined by the CLKDakDIV.CLKDakDIV[8:0] bits, where

<R> N = CLKDakDIV.CLKDakDIV[8:0]

The frequency of the output clock is calculated by

Output clock = Input clock / N

A new divisor value written to the CLKDakDIV.CLKDakDIV[8:0] bits becomes effective after the wait time below elapses.

Wait time The wait time is calculated by

Wait time = 20 x T_X x 20 x T_Y + 1 x T_Z

where

T_X: PCLK of IISBRGn selected by CKSC_102

T_Y: IISBRGn input clock

IISBRG00: Selected CKSC_110 clock

IISBRG01: Divided IISBRG00 clock

IISBRG10: Selected CKSC_115 clock

IISBRG11: Divided IISBRG10 clock

IISBRG20: Selected CKSC_116 clock

IISBRG21: Divided IISBRG20 clock

T_Z: Divided clock before applying new setting

The clock generated by IISBRG00 can be supplied as IISAACKO to provide the audio clock for external devices without their own clock generators.

(3) Selecting the serial clock for each channel

Use the SCKISL0IISA and SCKISL1IISA registers to select the serial clock supplied to each channel.

27.1.3 IISA baudrate generator register overview

IISA baudrate generators are controlled and operated by the following registers:

Table 27-8 IISA baudrate generator register overview

Register name	Symbol	Address
BRG00 divisor register	CLKD00DIV	FF82 9000_H
BRG01 divisor register	CLKD01DIV	FF82 9100_H
BRG10 divider register	CLKD10DIV	FF82 9200_H
BRG11 divider register	CLKD11DIV	FF82 9300_H
BRG20 divisor register	CLKD20DIV	FF82 9400_H
BRG21 divider register	CLKD21DIV	FF82 9500_H

27.1.4 IISA baudrate generator control register details

(1) CLKDakDIV – Divisor register

This register is used to define the clock divisors.

Access This register can be read or written in 16-bit units.

<R> **Address** CLKD00DIV: FF82 9000H, CLKD01DIV: FF82 9100H,
CLKD10DIV: FF82 9200H, CLKD11DIV: FF82 9300H,
CLKD20DIV: FF82 9400H, CLKD21DIV: FF82 9500H

Initial value 0000H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							CLKDakDIV[8:0]								
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

<R> **Table 27-9 CLKDakDIV register contents**

Bit position	Bit name	Function
8 to 0	CLKDak DIV[8:0]	Clock divisor N 000_H: N = no output 001_H: N = 1 002_H: N = 2 ... $1FE_H$: N = 510 $1FF_H$: N = 511

27.1.5 IISA serial clock selection register details

(1) SCKISLkIISA – Serial clock selection register

This register is used to select the serial clock.

Access This register can be read in 8-bit units.

Address SCKISL0IISA: FF77 2020H, SCKISL1IISA: FF77 2024H

Initial Value 00_H. This register is initialized by any reset.

	6	5	4	3	2	1	0
0	0	SCKISLk IISA5	SCKISLk IISA4	SCKISLk IISA3	SCKISLk IISA2	SCKISLk IISA1	SCKISLk IISA0
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 27-10 SCKISLkIISA register contents

Bit position	Bit name	Function
5 to 0	SCKISLkIISA [5-0]	Select the serial clock used for each channel.

SCKISL1 IISAn	SCKISL0 IISAn	Selected clock
0	0	IISBRG01 output
0	1	IISBRG11 output
1	×	IISBRG21 output

27.2 Functional Overview

IISA controls the serial data transfer for the digital audio format.

Features summary

IISA has the following features:

- Master and slave mode
- Full duplex data transfer and half duplex data transfer
- Support of serial audio format and I^2S format (both formats MSB left aligned and LSB right aligned)
- Transfer of digital audio data in 8, 16, 18, 20, or 24 bit units
- Support of a wide range of sampling frequencies driven by internal or external clocks
- Configurable active level of the word select signal IISAnWS
- Separate FIFOs for the transmission and reception of data (max. 16 data entries each with 24 bits)
- Generation of interrupt requests depending on how full the FIFOs are
- Masking of interrupt requests

The following figure shows the main components of IISA.

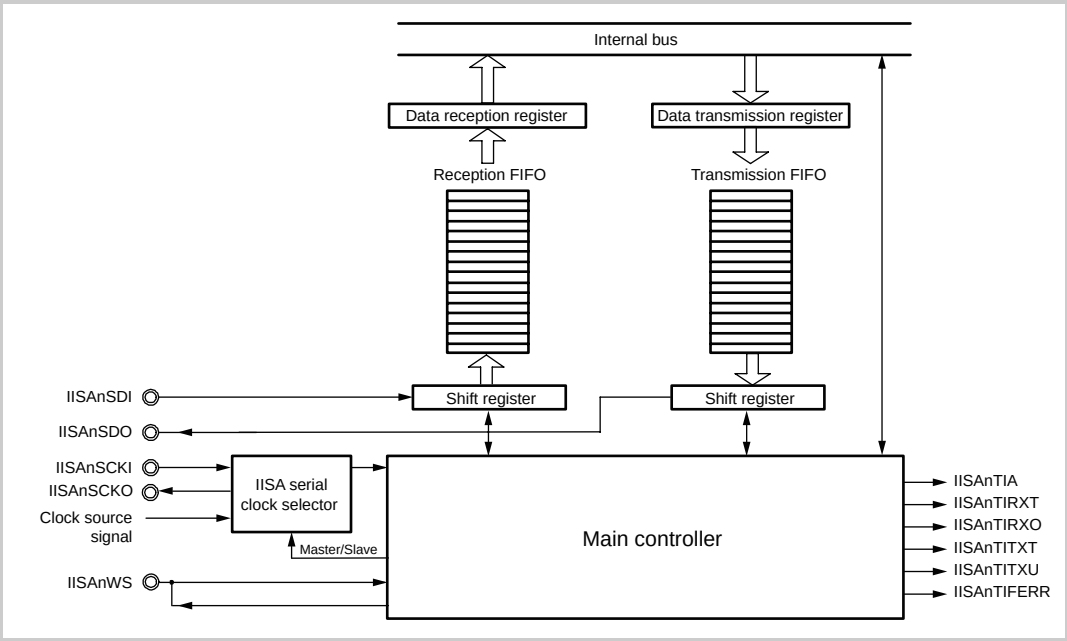


Figure 27-4 Block diagram of IISA with full-duplex data transfer

Serial clock selectors

For details about the serial clock selectors, see *27.1.1 "Serial clock selectors"*.

27.3 Functional Description

IISA performs data transmission/reception by using the serial data output/input IISAnSDO/IISAnSDI. It supports the serial audio format as well as the I^2S format.

- In full-duplex transfer mode, the digital audio data is transmitted in two directions at the same time using separate lines for IISAnSDO and IISAnSDI.
- In half-duplex transfer mode, IISAnSDO and IISAnSDI share one line.

Operation modes and clock selector

IISA can be operated in master and slave modes.

- The master mode allows an external and internal audio clock source to be used for the synchronization of the serial receive and transmit operations.
- In slave mode, the serial clock IISAnSCK and the word select signal IISAnWS are input from an external master.

For details about the configuration of the clocks, see *27.1.1 "Serial clock selectors"*.

Data reception/ transmission

IISA uses two separate FIFOs for the reception and transmission of data to optimize the transfer on IISA.

Word select signal

The digital audio data of the left and right channels is transferred in turn with the left channel's data first. The word select signal IISAnWS indicates which channel's data is being transferred. Depending on the operation mode, IISAnWS is generated by IISA or input from an external master.

Main controller

The main controller controls the shifting, formatting, and timing of the serial data, as well as the word select. It also contains all control and status registers of IISA.

Configuration

The active level for items such as the word select signal, data length, presentation of invalid data, alignment of data, and generation of interrupt requests can be configured as desired.

27.3.1 Operation modes

IISA can be operated in master and slave mode allowing different clock sources to be used.

For details, see *27.1.1 "Serial clock selectors"*.

The operation mode can be configured in IISAnCTL.IISAnMD.

- When operating as the master (IISAnCTL.IISAnMD bit = 1), IISA provides the serial clock IISAnSCK and the word select signal IISAnWS.
- In slave mode (IISAnCTL.IISAnMD bit = 0), IISAnSCK and IISAnWS are input from an external master.

27.3.2 Transmission and reception of data

(1) FIFOs

IISA is equipped with two independent FIFOs.

Each FIFO can cache up to 16 data entries each with up to 24 bits. However, one FIFO can hold up to 8 frames.

(2) Reception of data

The reception shift register changes the serial data received from the serial data input pin IISAnSDI into parallel data and stores it in the reception FIFO. From here the data is transferred to the 32-bit reception data register IISAnRX where it can be read.

The data is transferred at the rising edge of the serial clock IISAnSCK.

The data of the left and right channels is processed in turns. The first received data is data of the left channel.

(3) Transmission of data

Data to be transmitted is written in the transmission data buffer IISAnTX and then transferred to the transmission FIFO. The transmission shift register changes the parallel data of the transmission FIFO into serial data and outputs it to the serial data output pin IISAnSDO.

This operation is driven by the inverted serial clock (at the falling edge of IISAnSCK).

The data of the left and right channels is processed in turns. The data written to IISAnTX at first is data of the left channel.

Note When the transmission of data has been started (IISAnCTL.IISAnTXE bit = 1), the transmission FIFO must be completely filled before data output to IISAnSDO starts.

(4) Verification of the FIFO's fill level

IISA generates interrupt requests and sets status flags according to the fill level of the FIFOs. For example, to notify that a FIFO's fill level has reached a configurable threshold, or that a FIFO is full or empty. See *27.3.6 "Generation of interrupt requests and status flags" on page 2108* for details.

IISAnSTR1 indicates the number of unsent/unread data entries currently cached in the transmission and reception FIFOs.

27.3.3 Word select signal (IISAnWS)

The word select signal IISAnWS indicates which channel's data is being transferred. IISAnCTL.IISAnWSL specifies the active level that indicates the *left* channel's data.

- In slave mode (IISAnCTL.IISAnMD bit = 0), IISAnWS is input from an external master.
- When operating as the master (IISAnCTL.IISAnMD bit = 1), IISA provides IISAnWS.

  IISAnWS is immediately output after IISA data transmission and/or reception has been started (IISAnCTL.IISAnTXE and/or IISAnCTL.IISAnRXE have been set to 1).

Note A framing error occurs when IISAnWS does not match the number of serial clock cycles per frame set in IISAnCTL.IISAnSLFS[1:0].

See *(2) "Framing error" on page 2109* for details.

27.3.4 Data length and data format

(1) Data length

IISA transfers digital audio data in 8, 16, 18, 20, or 24 bit units. Considering the data of the left and right channels, up to 48 bits per frame can be processed.

The data length can be configured in IISAnCTL.IISAnDLG[4:0].

Note The data length must be set *before* writing audio data to the transmission data register IISAnTX.

(2) Serial clock cycles per frame

The digital audio data of the left and right channels is transferred along the same serial transmission line in turn on the basis of time division multiplexing: Time slots that correspond to the configured serial clock cycles per frame (IISAnCTL.IISAnSLFS[1:0]) are reserved for the audio data of one frame.

Notes

1. The configured serial clock cycles per frame must be at least twice as long as the data length. Otherwise the data transfer becomes undefined.

   IISAnCTL.IISAnSLFS[1:0] > IISAnCTL.IISAnDLG[4:0] x 2

2. IISAnCTL.IISAnSLFS[1:0] must be set according to the required serial clock frequency IISAnSCK (= IISAnCTL.IISAnSLFS[1:0] x Fs).

   See *27.1.1 "Serial clock selectors"* for how to configure the serial clock appropriately.

(3) Data format

Bit order

The serial bit stream of a channel has to be stored in a fixed order where the most significant bit (MSB) is expected to be the first bit and the least significant bit (LSB) the last bit.

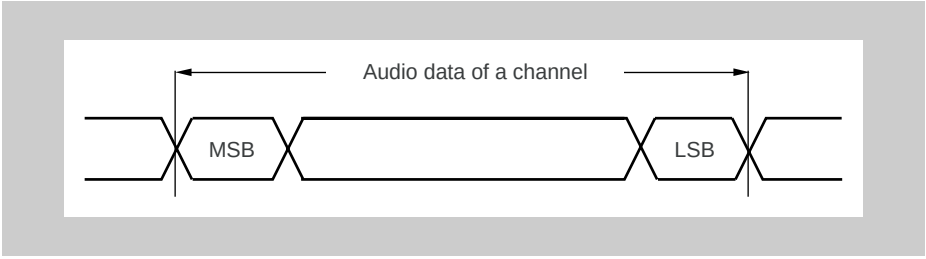


Figure 27-5 Bit order of audio data

Note Within one frame, the left channel's data is always first.

Presentation of odd/invalid bits

When IISAnCTL.IISAnSLFS[1:0] > IISAnCTL.IISAnDLG[4:0] x 2, the invalid bits are set to 0 or 1 as configured in IISAnCTL.IISAnIDL.

Bit alignment

The position of the invalid data compared to the audio data depends on the configured alignment (IISAnCTL.IISAnSLDF): MSB left aligned or LSB right aligned.

The following figures illustrate the different alignments.

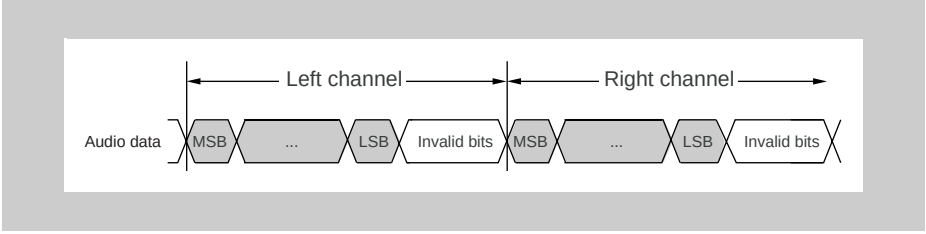


Figure 27-6 MSB left aligned (IISAnCTL.IISAnSLDF bit = 0)

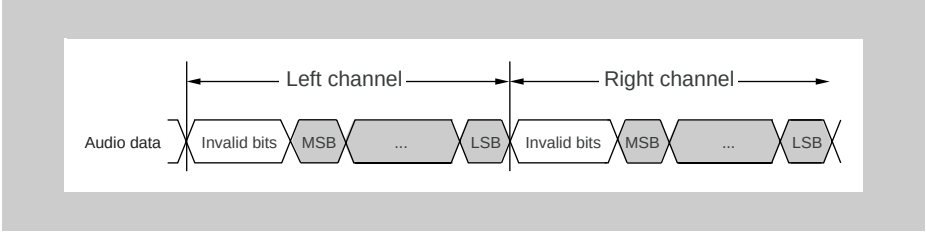


Figure 27-7 LSB right aligned (IISAnCTL.IISAnSLDF bit = 1)

27.3.5 Transfer format

IISA is compliant with the I^2S format and also supports the serial audio format.

The transfer format can be configured in IISAnCTL.IISAnSLTF.

- In the serial audio format (IISAnCTL.IISAnSLTF bit = 0), the MSB of a channel is expected/transmitted *at the same time* IISAnWS changes.
- In the I^2S format (IISAnCTL.IISAnSLTF bit = 1), the MSB of the next channel is expected/transmitted *one clock cycle after* IISAnWS changes.

Timing The following figures illustrate the transfer timing in both transfer formats.

The word select of the left channel is active low (IISAnCTL.IISAnWSL bit = 0) and the MSB is left aligned (IISAnCTL.IISAnSLDF bit = 0)

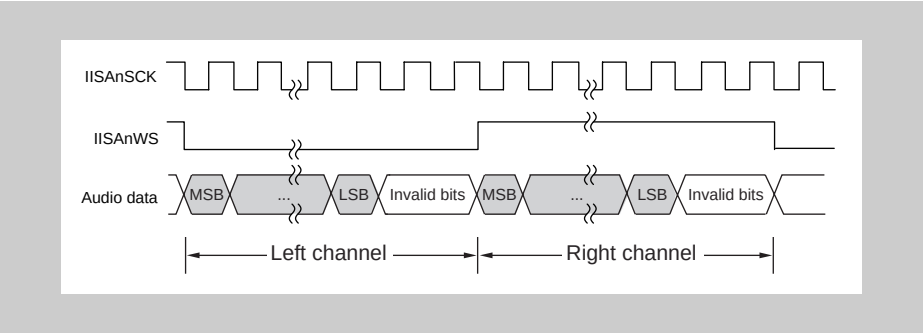


Figure 27-8 Transfer timing in serial audio format (IISAnCTL.IISAnSLTF bit = 0)

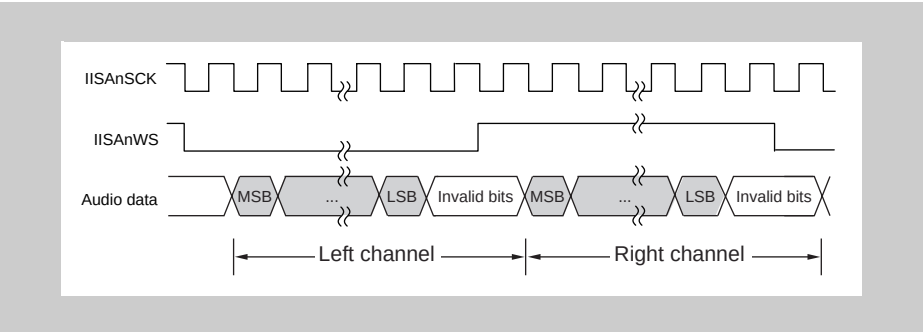


Figure 27-9 Transfer timing in I^2S transfer format (IISAnCTL.IISAnSLTF bit = 1)

27.3.6 Generation of interrupt requests and status flags

Interrupt requests IISA generates the general interrupt request IISAnTIA for any of the following conditions:

- The fill level of the reception FIFO has reached the configured threshold
- The reception FIFO is full (overrun)
- The fill level of the transmission FIFO has reached the configured threshold
- The transmission FIFO is empty (underrun)
- The configured serial clock cycles per frame do not agree with the input words select signal IISAnWS (framing error). This condition can only occur in slave mode.

Note If multiple conditions are true, the general interrupt request IISAnTIA is generated only *once*.

IISA additionally generates a specific interrupt request if configured in IISAnOPT for the corresponding condition.

Note The general interrupt request IISAnTIA is generated when any of the above conditions are true and cannot be masked in IISAnOPT. IISAnOPT is used to mask additional interrupt requests that depend on a certain condition.

Status flags The status of the FIFOs can be checked in the IISAnSTR0 register where status flags are set according to the conditions listed above.

Note Specific interrupt requests can be masked in IISAnOPT. Therefore, IISAnSTR0 should be used to verify the source of the general interrupt IISAnTIA.

(1) Indication of the FIFO's fill level

The following table provides an overview about the interrupt request and status flags that indicate the FIFO's fill level.

Table 27-11 Indication of the FIFO's fill level

	Status/fill level	Interrupt request	Status flag in IISAnSTR0
Reception FIFO	FIFO caches number of data entries (0 to 16) configured in IISAnOPT.RTGS[3:0].	• IISAnTIA • IISAnTIRXT when IISAnOPT.RTM bit = 0	• IISAnRTF
	Overrun[a]: FIFO is full and the last written data entry is overwritten by following data.	• IISAnTIA • IISAnTIRXO when IISAnOPT.ROM bit = 0	• IISAnROE
	FIFO is empty.		• IISAnREF
Transmission FIFO	FIFO caches number of data entries (0 to 16) configured in IISAnOPT.TTGS[3:0].	• IISAnTIA • IISAnTITXT when IISAnOPT.TTM bit = 0	• IISAnTTF
	Underrun[a]: FIFO is empty. Invalid data (as specified in IISAnCTL.IISAnIDL) is written to the transmission shift register and output to IISAnSDO.	• IISAnTIA • IISAnTITXU when IISAnOPT.TUM bit = 0	• IISAnTUE
	FIFO is full. It caches 16 data entries or 8 frames.		• IISAnTFF

a) The data transfer needs to be restarted after a FIFO overrun/underrun. See *(3) "Restarting data transfer after FIFO overrun and/or underrun" on page 2112* for details.

Note The FIFO's fill level is checked whenever IISAnOPT is written and when new data has been added to the reception FIFO or data has been transferred from the transmission FIFO to the transmission shift register.

(2) Framing error

A framing error occurs when the word select signal IISAnWS does not match the number of serial clock cycles per frame set in IISAnCTL.IISAnSLFS[1:0].

A framing error can only occur in slave mode (IISAnCTL.IISAnMD bit = 0) where IISAnWS is input from an external master.

Interrupt requests and flags

If a framing error occurs:

- The general interrupt request IISAnTIA is generated.
- The interrupt request IISAnTIFERR is generated when IISAnOPT.FEM bit = 0.
- The interrupt status flag bit IISAnSTR0.IISAnFE is set to 1.

Presentation of invalid bits

For data reception: when IISAnWS changes before all of the expected data has been received, the invalid bits are either set to “0” or “1” as specified in IISAnCTL.IISAnIDL.

Example

The following figure illustrates a framing error.

The conditions are as follows:

- IISAnWS is input from a master (IISAnCTL.IISAnMD bit = 0) and the word select for the left channel is active low (IISAnCTL.IISAnWSL bit = 0).
- 32 serial clock cycles per frame is selected (IISAnCTL.IISAnSLF[1:0] bits = 00_B).
- The data length is 8 bits per channel (IISAnCTL.IISAnDLG[4:0] bits = 01000_B).
- The MSB left-aligned format is selected (IISAnCTL.IISAnSLDF bit = 0).

According to these settings, a slot of 16 serial clock cycles is reserved for a channel: 8 audio data bits are followed by 8 bits which are set to the configured invalid value (IISAnCTL.IISAnIDL).

When IISAnWS switches before all data of a channel (according to the configured serial clock cycles per frame) is received/transmitted, IISA indicates the framing error by the means listed above.

For data reception, the bits that follow the framing error are invalid.

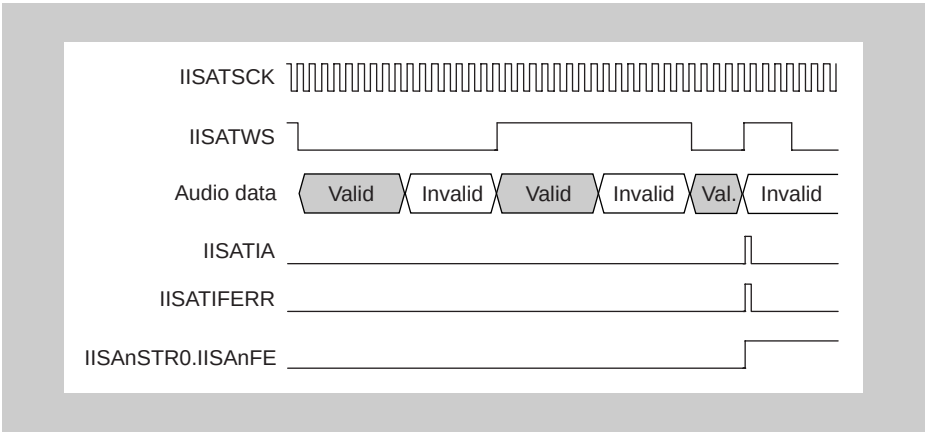


Figure 27-10 Framing error when receiving data

27.3.7 Basic procedures

This section describes the basic procedures for the usage of IISA.

(1) Starting operation

To start the operation of IISA:

1. Enable an appropriate clock supply for the intended operation mode and required sampling frequency:
   See *27.1.1 "Serial clock selectors"* for details.
2. Configure the operation mode and data transfer in IISAnCTL as desired.
3. Specify the trigger level for the generation of interrupt requests for the transmission FIFO (IISAnOPT.IISAnTTGS[3:0]) and the reception FIFO (IISAnOPT.IISAnRTGS[3:0]).
4. Initialize the FIFOs by writing 1 to IISAnSTC.IISAnCLTP and IISAnSTC.IISAnCLRP.
5. Start the transmission and/or reception of data by writing 1 to IISAnCTL.IISAnTXE and/or IISAnCTL.IISAnRXE.

   The data transfer starts at the edge of the next word select. The left channel's data is transferred first.
   See *27.3.2 "Transmission and reception of data" on page 2104* for details.

Note IISAnCTL must not be changed when the transmission/reception of data is enabled, except for disabling the transmission/reception of data by setting IISAnCTL.IISAnTXE or IISAnCTL.IISAnRXE to 0.

(2) Stopping operation

To stop the operation of IISA:

- Stop the transmission and/or reception of data by writing 0 to IISAnCTL.IISAnTXE and/or IISAnCTL.IISAnRXE.
  - Transmission of data is stopped at the next edge of the word select signal IISAnWS or — in other words — when the transmission of the current channel's data has finished.
    If the stop is triggered while the left channel's data is being transmitted, this channel's data is transmitted correctly, but the corresponding data of the right channel will be invalid.
  - Reception of data is stopped at the edge of the next serial clock IISAnSCK.

Note In master mode, IISA stops outputting the serial clock IISAnSCK and the word select signal IISAnWS as soon as the serial clock input from the clock selector has been stopped.

(3) Restarting data transfer after FIFO overrun and/or underrun

The data transfer has to be restarted when the transmission FIFO cannot provide data because it is empty (underrun) or the reception FIFO is full and cannot cache new data (overrun).

See *(1) "Indication of the FIFO's fill level" on page 2109* for details.

To restart the data transfer:

1. Stop the transmission and/or reception of data by writing 0 to IISAnCTL.IISAnTXE and/or IISAnCTL.IISAnRXE.
2. Initialize the FIFOs by writing 1 to IISAnSTC.IISAnCLTP and/or IISAnSTC.IISAnCLRP.
3. Start the transmission and/or reception of data by writing 1 to IISAnCTL.IISAnTXE and/or IISAnCTL.IISAnRXE.

The transfer of data starts with the left channel's data first.

27.4 Registers

This section contains a description of all the registers of IISA.

27.4.1 IISA register overview

IISA is controlled and operated by the following registers:

Table 27-12 IISA register overview

Register name	Symbol	Address
Control registers		
IISA control register	IISAnCTL	<IISAn_base>
IISA option register	IISAnOPT	<IISAn_base> + 04_H
Status registers		
IISA status register 0	IISAnSTR0	<IISAn_base> + 08_H
IISA status register 1	IISAnSTR1	<IISAn_base> + $0C_H$
IISA status clear register	IISAnSTC	<IISAn_base> + 10_H
Data registers		
IISA reception data register	IISAnRX	<IISAn_base> + 14_H
IISA transmission data register	IISAnTX	<IISAn_base> + 18_H

<IISAn_base> The base addresses <IISAn_base> of IISAn are defined in the first section of this chapter under the key word “Register addresses”.

27.4.2 Control register details

(1) IISAnCTL – IISA control register

This register is used to control the general behavior of IISA.

Access This register can be read/written in 32-bit units. It must not be changed during ongoing data transfers, except for disabling the transmission/reception of data (IISAnCTL.IISAnTXE and/or IISAnCTL.IISAnRXE = 0).

Address <IISAn_base>

Initial Value 0000 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	IISAn SLFS[1:0]		IISAnDLG[4:0]					IISAn MD	IISAn IDL	IISAn SLTF	IISAn WSL	IISAn SLDF	IISAn TXE	IISAn RXE	0
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R

Table 27-13 IISAnCTL register contents (1/2)

Bit position	Bit name	Function
14, 13	IISAnSLFS[1:0]	Specifies the number of serial clock cycles in 1 frame: IISAn SLFS1 / IISAn SLFS0 / Serial clock cycles in 1 frame 0 / 0 / 32 0 / 1 / 48 1 / 0 / 64 1 / 1 / 128 See *(2) "Serial clock cycles per frame" on page 2105* for details. IISAnCTL.IISAnSLFS[1:0] must be at least twice as long as the data length (IISAnCTL.IISAnDLG[4:0] x 2). Otherwise the data transfer becomes undefined.
12 to 8	IISAnDLG[4:0]	Specifies the data length: IISAn DLG4 / IISAn DLG3 / IISAn DLG2 / IISAn DLG1 / IISAn DLG0 / Data length 0 / 1 / 0 / 0 / 0 / 8 bits 1 / 0 / 0 / 0 / 0 / 16 bits 1 / 0 / 0 / 1 / 0 / 18 bits 1 / 0 / 1 / 0 / 0 / 20 bits 1 / 1 / 0 / 0 / 0 / 24 bits Other settings are prohibited.
7	IISAnMD	Specifies the operation mode: 0: Slave mode 1: Master mode See *27.3.1 "Operation modes" on page 2104* for details.

Table 27-13 IISAnCTL register contents (2/2)

Bit position	Bit name	Function
6	IISAnIDL	Specifies the invalid data during data transmission/reception: 0: Output '0' as invalid data 1: Output '1' as invalid data See the corresponding section in *(3) "Data format" on page 2106* for details.
5	IISAnSLTF	Specifies the transfer format: 0: Serial audio format 1: I^2S format See the corresponding section in *27.3.5 "Transfer format" on page 2107* for details.
4	IISAnWSL	Specifies the active level of IISAnWS when left channel's data is transferred: 0: Active low 1: Active high See the corresponding section in *27.3.5 "Transfer format" on page 2107* for details.
3	IISAnSLDF	Specifies the bit alignment of a channel's data: 0: MSB left aligned 1: LSB right aligned See the corresponding section in *(3) "Data format" on page 2106* for details.
2	IISAnTXE	Enables/disables the transmission of data: 0: Disabled, no data is output to IISAnSDO 1: Enabled
1	IISAnRXE	Enables/disables the reception of data: 0: Disabled, input data from IISAnSDI is ignored/discarded 1: Enabled

<R> **Caution** The IISAnCTL register setting takes effect after (6 × PCLK + 6 × IISA0SCK) cycles have elapsed. The IISAnCTL register cannot be written while the PCLK and IISA0SCK clocks are not being supplied.

(2) IISAnOPT – IISA option register

This register specifies the generation of interrupt requests.

See *27.3.6 "Generation of interrupt requests and status flags" on page 2108* for details.

Access This register can be read/written in 32-bit units. It must not be changed during ongoing data transfers, for example if IISAnCTL.IISAnTXE bit = 1 and/or IISAnCTL.IISAnRXE bit = 1.

Address <IISAn_base> + 04_H

Initial Value $0000\ 001F_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IISAnTTGS[3:0]				IISAnRTGS[3:0]				0	0	0	IISAn FEM	IISAn TUM	IISAn TTM	IISAn ROM	IISAn RTM
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 27-14 IISAnOPT register contents (1/2)

Bit position	Bit name	Function
15 to 12	IISAnTTGS[3:0]	Specifies at which fill level of the transmission FIFO an interrupt request is generated: IISAn TTGS3 / IISAn TTGS2 / IISAn TTGS1 / IISAn TTGS0 / No. of data entries in transmission FIFO 0 / 0 / 0 / 0 / Transmission FIFO empty 0 / 0 / 0 / 1 / 1 0 / 0 / 1 / 0 / 2 0 / 0 / 1 / 1 / 3 0 / 1 / 0 / 0 / 4 ... / ... / ... / ... / ... 1 / 1 / 1 / 1 / 15 See *(1) "Indication of the FIFO's fill level" on page 2109* for details.

Table 27-14 IISAnOPT register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>11 to 8</td><td>IISAnRTGS[3:0]</td><td>Specifies at which fill level of the reception FIFO an interrupt request is generated:
<table>
<tr><th>IISAn RTGS3</th><th>IISAn RTGS2</th><th>IISAn RTGS1</th><th>IISAn RTGS0</th><th>No. of data entries in reception FIFO</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Reception FIFO full (caches 16 data entries)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>15</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>14</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>13</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>12</td></tr>
<tr><td colspan="4">...</td><td>...</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
See *(1) “Indication of the FIFO’s fill level” on page 2109* for details.</td></tr>
<tr><td>4</td><td>IISAnFEM</td><td>Enables/disables the generation of the interrupt IISAnTIFERR in case of a framing error:
0: Generate interrupt request
1: Do not generate interrupt request</td></tr>
<tr><td>3</td><td>IISAnTUM</td><td>Enables/disables the generation of the interrupt IISAnTITXU in case of a transmission FIFO underrun:
0: Generate interrupt request
1: Do not generate interrupt request</td></tr>
<tr><td>2</td><td>IISAnTTM</td><td>Enables/disables the generation of the interrupt request IISAnTITXT in case the transmission FIFO’s fill level has reached the configured threshold (IISAnOPT.IISAnTTGS[3:0]):
0: Generate interrupt request
1: Do not generate interrupt request</td></tr>
<tr><td>1</td><td>IISAnROM</td><td>Enables/disables the generation of the interrupt request IISAnTIRXO if a reception FIFO overrun occurs:
0: Generate interrupt request
1: Do not generate interrupt request</td></tr>
<tr><td>0</td><td>IISAnRTM</td><td>Enables/disables the generation of the interrupt request IISAnTIRXT if the reception FIFO’s fill level has reached the configured threshold (IISAnOPT.IISAnRTGS[3:0]):
0: Generate interrupt request
1: Do not generate interrupt request</td></tr>
</table>

27.4.3 Status register details

(1) IISAnSTR0 – IISA status register 0

This register indicates the FIFO status and the interrupt request status.

The status flags in this register are set no matter whether the generation of an interrupt request is masked in IISAnOPT or not. Therefore, IISAnSTR0 should be used to verify the source of the general interrupt IISAnTIA.

See *27.3.6 “Generation of interrupt requests and status flags” on page 2108* for details.

Access This register is read-onlr, in 32-bit units.

Address <IISAn_base> + 08_H

Initial Value $0000\ 0001_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	IISAn FE	IISAn TUE	IISAn TTF	IISAn ROE	IISAn RTF	IISAn TFF	IISAn REF
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 27-15 IISAnSTR0 register contents (1/2)

Bit position	Bit name	Function
6	IISAnFE	Indicates a framing error: 0: No framing error 1: Framing error This bit is cleared when IISAnSTC.IISAnFEC is set to 1.
5	IISAnTUE	Indicates a transmission FIFO underrun: 0: No underrun 1: Underrun This bit is cleared when IISAnSTC.IISAnTUC is set to 1.
4	IISAnTTF	Indicates that the fill level of the transmission FIFO has reached the configured threshold: 0: Transmission FIFO contains more data entries than configured threshold 1: Threshold reached This bit is cleared when IISAnSTC.IISAnTTC is set to 1.
3	IISAnROE	Indicates a reception FIFO overrun: 0: No overrun 1: Overrun This bit is cleared when IISAnSTC.IISAnROC is set to 1.

Table 27-15 IISAnSTR0 register contents (2/2)

Bit position	Bit name	Function
2	IISAnRTF	Indicates that the fill level of the reception FIFO has reached the configured threshold: 0: Reception FIFO contains less data entries than configured threshold 1: Threshold reached This bit is cleared when IISAnSTC.IISAnRTC is set to 1.
1	IISAnTFF	Specifies whether the transmission FIFO is full: 0: Not full 1: Full (caches 8 frames) This bit is cleared when IISAnSTC.IISAnCLTP is set to 1.
0	IISAnREF	Specifies whether the reception FIFO is empty: 0: Not empty 1: Empty This bit is cleared when IISAnSTC.IISAnCLRP is set to 1.

(2) IISAnSTC – IISA status clear register

This register is the clear control register of IISAnSTR0. It is also used to initialize the FIFOs.

Access This register can be read/written in 32-bit units.

Address <IISAn_base> + 10_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	IISAn FEC	IISAn TUC	IISAn TTC	IISAn ROC	IISAn RTC	0	0	0	0	0	0	IISAn CLTP	IISAn CLRP
R	R	R	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R	R	R/W	R/W

Table 27-16 IISAnSTC register contents

Bit position	Bit name	Function
12	IISAnFEC	0: No function 1: Clears the status flag IISAnSTR0.IISAnFE that indicates a framing error
11	IISAnTUC	0: No function 1: Clears the status flag IISAnSTR0.IISAnTUE that indicates an underrun of the transmission FIFO
10	IISAnTTC	0: No function 1: Clears the status flag IISAnSTR0.IISAnTTF that indicates that the fill level of the transmission FIFO has reached the configured threshold
9	IISAnROC	0: No function 1: Clears the status flag IISAnSTR0.IISAnROE that indicates an overrun of the reception FIFO
8	IISAnRTC	0: No function 1: Clears the status flag IISAnSTR0.IISAnRTF that indicates that the fill level of the reception FIFO has reached the configured threshold
1	IISAnCLTP	0: No function 1: Initializes the transmisisson FIFO: - Transmission FIFO is cleared - Pointer of the transmission FIFO is reset - Status flag IISAnSTR0.IISAnTFF is cleared
0	IISAnCLRP	0: No function 1: Initializes the reception FIFO: - Reception FIFO is cleared - Pointer of the reception FIFO is reset - Status flag IISAnSTR0.IISAnREF is cleared

<R> **Caution** The IISAnCTL register setting takes effect after (6 × PCLK + 6 × IISA0SCK) cycles have elapsed. The IISAnCTL register cannot be written while the PCLK and IISA0SCK clocks are not being supplied.

(3) IISAnSTR1 – IISA status register 1

This register indicates the number of data entries cached in the FIFOs. Each FIFO can cache a maximum of 16 data entries or 8 frames.

Access This register is read-only, in 32-bit units.

Address <IISAn_base> + $0C_H$

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	IISAnTPF[4:0]					0	0	0	IISAnRPF[4:0]				
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 27-17 IISAnSTR1 register contents

Bit position	Bit name	Function
12 to 8	IISAnTPF[4:0]	Indicates the number of unsent data entries cached in the transmission FIFO: 00000_B to 10000_B
4 to 0	IISAnRPF[4:0]	Indicates the number of unread data entries cached in the reception FIFO: 00000_B to 10000_B

27.4.4 Data register details

(1) IISAnRX – IISA reception data register

This register is used to read received data that is cached in the reception FIFO. See *(2) "Reception of data" on page 2104* for details.

Access This register is read-only, in 32-bit units.

Address <IISAn_base> + 14_H

Initial Value 0000 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	IISAnRX[23:16]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IISAnRX[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 27-18 IISAnRX register contents

Bit position	Bit name	Function
23 to 0	IISAnRX[23:0]	Contains one channel's data that has been received The left and right channels of the reception FIFO are read in turn – with the data of the left channel first.

(2) IISAnTX – IISA transmission data register

This register is used to write data to be transmitted to the transmission FIFO.

See *(3) "Transmission of data" on page 2104* for details.

Access This register can be read/written in 32-bit units.

Address <IISAn_base> + 18_H

Initial Value 0000 0000_H. This register is initialized by any reset.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	0	0	0	0	0	0	0	0	IISAnTX[23:16]							
<R>	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	IISAnTX[15:0]															
<R>	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 27-19 IISAnTX register contents

Bit position	Bit name	Function
23 to 0	IISAnTX[23:0]	Contains one channel's data that is to be transmitted The left and right channels of the transmission FIFO are written in turn – with the data of the left channel first.

Chapter 28 PCM Interface (PCM)

This chapter describes PCM interface (PCM).

The first section describes all V850E2/Sx4-H specific properties, such as instances, register base addresses, and input/output signal names.

The subsequent sections describe the features that apply to all implementations.

28.1 V850E2/Sx4-H PCM Features

Instances

This microcontroller has following number of instances of PCM:

Table 28-1 Instances of PCM

PCMn	V850E2/SG4-H	V850E2/SJ4-H	V850E2/SK4-H
Number of instances	1	2	
Name	PCM0	PCM0, PCM1	

Instances index n

Throughout this chapter, the individual instances of PCM is identified by the index "n" (n = 0, 1), for example, PCMnMOD for the operation mode setting register.

Register addresses

All PCMn register addresses are given as addresses offset from the individual base address <PCMn_base>.

The base address <PCMn_base> of each PCMn is listed in the following table:

Table 28-2 Register base addresses <PCMn_base>

PCMn	<PCMn_base> address
PCM0	FF83 3000_H
PCM1	FF83 3800_H

Clock supply

The following clock is input to PCM:

Table 28-3 PCM clock supply

PCMn	Clock	Connected to:
PCM0	PCM0SCK	PCM0 serial clock selector
	PCLK	Clock generator CKSCLK_122
PCM1	PCM1SCK	PCM1 serial clock selector
	PCLK	Clock generator CKSCLK_123

Interrupts and DMA PCMn can generate the following interrupt and DMA requests:

Table 28-4 PCMn interrupt and DMA requests (1/2)

PCMn signal	Function	Connected to:
PCM0:		
PCM0INT	General interrupt	Interrupt controller INTPM0INT
PCM0RXFRE	Reception frame synchronization error interrupt	Interrupt controller INTPM0RXFRE
PCM0RXORE	Reception overrun error interrupt	Interrupt controller INTPM0RXORE
PCM0RXREN	Reception data FIFO interrupt	Interrupt controller INTPM0RXREN
PCM0RXURE	Reception underrun error interrupt	Interrupt controller INTPM0RXURE
PCM0TXFRE	Transmission frame synchronization error interrupt	Interrupt controller INTPM0TXFRE
PCM0TXORE	Transmission overrun error interrupt	Interrupt controller INTPM0TXORE
PCM0TXWEN	Transmission data FIFO interrupt	Interrupt controller INTPM0TXWEN
PCM0TXURE	Transmission underrun error interrupt	Interrupt controller INTPM0TXURE
PCM0TDMARQ	DMA transmission trigger	DMA controller trigger 116 PM0TDMARQ
PCM0RDMARQ	DMA reception trigger	DMA controller trigger 119 PM0RDMARQ
PCM1:		
PCM1INT	General interrupt	Interrupt controller INTPM1INT
PCM1RXFRE	Reception frame synchronization error interrupt	Interrupt controller INTPM1RXFRE
PCM1RXORE	Reception overrun error interrupt	Interrupt controller INTPM1RXORE
PCM1RXREN	Reception data FIFO interrupt	Interrupt controller INTPM1RXREN
PCM1RXURE	Reception underrun error interrupt	Interrupt controller INTPM1RXURE
PCM1TXFRE	Transmission frame synchronization error interrupt	Interrupt controller INTPM1TXFRE
PCM1TXORE	Transmission overrun error interrupt	Interrupt controller INTPM1TXORE
PCM1TXWEN	Transmission data FIFO interrupt	Interrupt controller INTPM1TXWEN
PCM1TXURE	Transmission underrun error interrupt	Interrupt controller INTPM1TXURE

Table 28-4 PCMn interrupt and DMA requests (2/2)

PCMn signal	Function	Connected to:
PCM1TDMARQ	DMA transmission trigger	DMA controller trigger 120 PM1TDMARQ
PCM1RDMARQ	DMA reception trigger	DMA controller trigger 121 PM1RDMARQ

PCM hardware reset

PCM and its registers are initialized by the following reset signal:

Table 28-5 PCMn reset signal

PCMn	Reset signal
PCMn	System reset SYSRES

I/O signals

The I/O signals of PCMn are listed in the following table:

Table 28-6 PCMn I/O signals

PCMn signal	Function	Connected to:
PCM0:		
PCM0CLK	PCM0 clock I/O	Port PCM0CLK
PCM0SEN	PCM0 frame synchronization I/O	Port PCM0SEN
PCM0SI	PCM0 data input	Port PCM0SI
PCM0SO	PCM0 data output	Port PCM0SO
PCM1:		
PCM1CLK	PCM1 clock I/O	Port PCM1CLK
PCM1SEN	PCM1 frame synchronization I/O	Port PCM1SEN
PCM1SI	PCM1 data input	Port PCM1SI
PCM1SO	PCM1 data output	Port PCM1SO

28.1.1 Serial clock selector

PCM uses the serial clock PCMnSCK as the reference clock for all reception and transmission operations.

PCM can be operated in master and slave mode allowing different clock sources to be used. The clock selector provides the correct clocks to be used.

(1) Serial clock signals in master mode

In master mode (PCMnMOD.M_S[1:0] bits = 01), PCM can generate the audio clock IISAACK and the serial clock PCMnSCK by selecting and dividing an external or internal clock source signal.

- IISAACK can be output as IISAACKO to provide the audio system clock for devices without their own clock generation. IISAACK is also used by the IISA interface.
- PCMnSCK is output as PCMnCLKO allowing slaves to synchronize with PCM.

Sampling frequency

The sampling frequency (Fs) is the frequency of the word select signal PCMnSEN. It depends on the audio system clock and the serial clock.

Typically, the clock source must be scaled down to achieve the required sampling frequency (Fs), audio clock IISAACK (= a x Fs), and serial clock PCMnSCK (= s x Fs).

Selecting and downscaling the clock source signal is done by the PCM master clock generator.

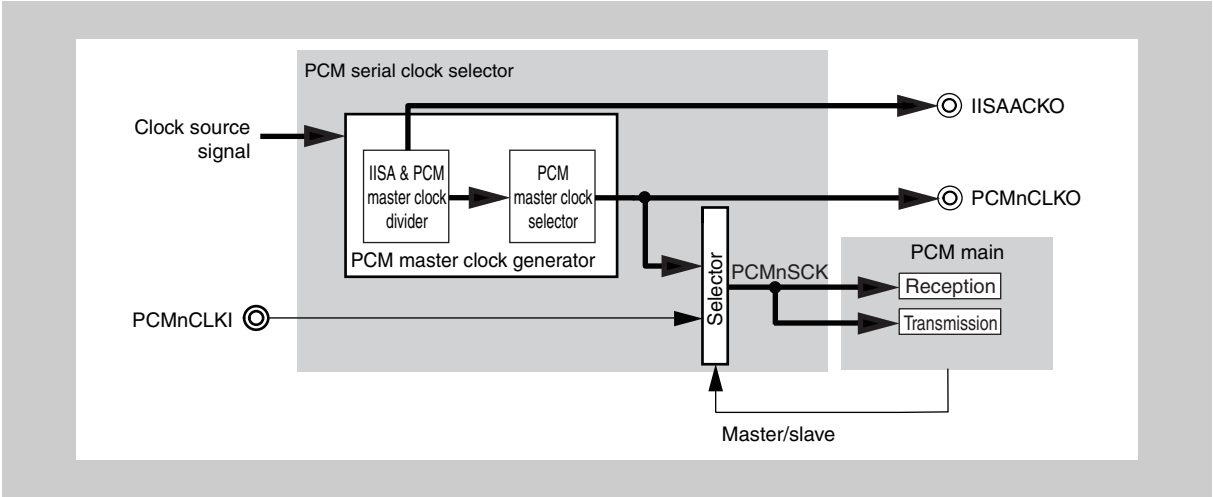


Figure 28-1 Serial clock signals in master mode

(2) Serial clock signals in slave mode

In slave operation mode (PCMnMOD.M_S[1:0] bits = 10), the serial clock PCMnSCK is driven by the external device pin PCMnCLKI. The audio clock IISAACK is not used.

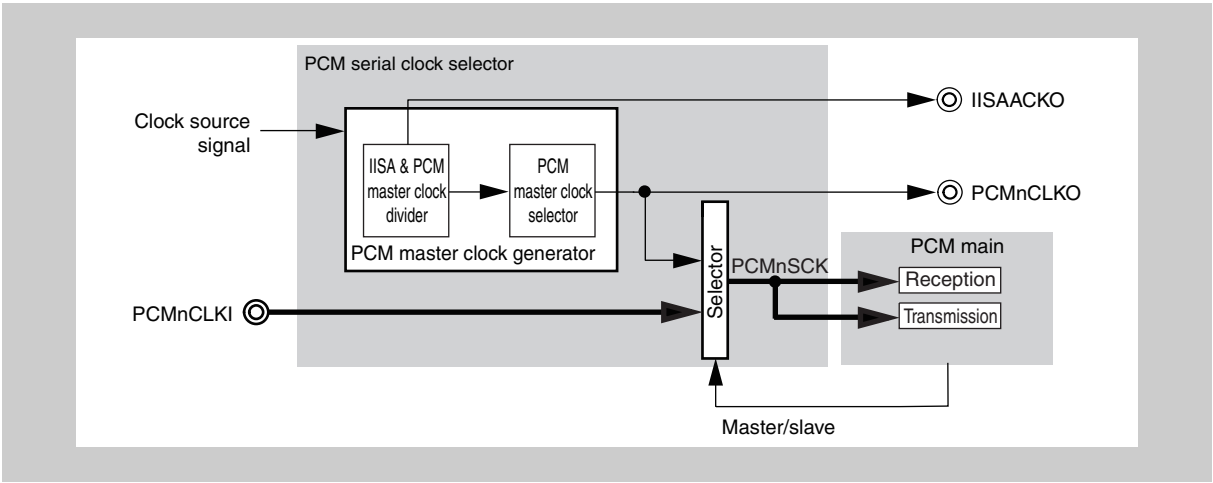


Figure 28-2 Serial clock signals in slave mode

28.1.2 PCM master clock generator

In master mode (PCMnMOD.M_S[1:0] bits = 01), PCM can generate the audio clock IISAACK and the serial clock PCMnSCK by selecting and dividing an external or internal clock source signal.

The IISA and PCM master clock dividers, which are included in the PCM master clock generator, can also be used to generate the IISA master clock.

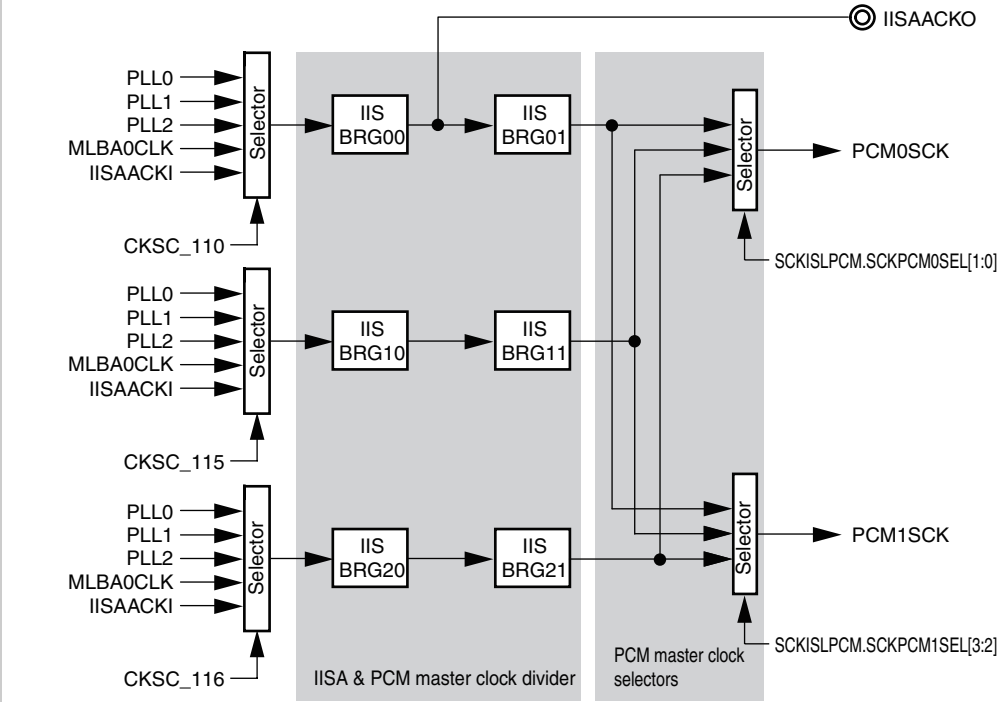


Figure 28-3 PCM master clock generator

(1) Source clock signals and baudrate generators

The clock source signal used by the PCM master clock generator and the baudrate generators is also used by IISA. For details about selecting the clock source signal and setting the baudrate generators, see *27.1.2 "IISA master clock generator"*.

(2) Selecting the serial clock for each channel

Use the SCKISLPCM register to select the serial clock supplied to each channel.

28.1.3 PCM serial clock selection register details

(1) SCKISLPCM - Serial clock selection register

The SCKISLPCM register is used to select the serial clock.

Access This register can be read or written in 8-bit units.

Address FF77 202C$_H$

Initial value 00$_H$. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	SCKPCM1 SEL3	SCKPCM1 SEL2	SCKPCM0 SEL1	SCKPCM0 SEL0
R	R	R	R	R/W	R/W	R/W	R/W

Table 28-7 SCKISLPCM register contents

Bit position	Bit name	Function
3, 2	SCKPCM1 SEL[3:2]	Select the serial clock used by PCM1. SCKPCM1 SEL3 / SCKPCM1 SEL2 / Selected clock 0 / 0 / IISBRG01 output 0 / 1 / IISBRG11 output 1 / × / IISBRG21 output
1, 0	SCKPCM0 SEL[1:0]	Select the serial clock used by PCM0. SCKPCM0 SEL1 / SCKPCM0 SEL0 / Selected clock 0 / 0 / IISBRG01 output 0 / 1 / IISBRG11 output 1 / × / IISBRG21 output

28.2 Functional Overview

Features summary

PCMn has the following features:

- Pin configuration
  - PCMnCLK: Serial clock I/O
  - PCMnSI: Serial data input
  - PCMnSO: Serial data output
  - PCMnSEN: Frame synchronization signal
- Serial interface for voice/audio codec
- Seven types of operating modes that can be used with various serial interfaces.
  - Monaural mode (modes 0 and 1)
  - I^2S mode (mode 2)
  - MSB left-justified mode (mode 3)
  - LSB right-justified mode (mode 4)
  - TDM mode (modes 5 and 6)
- Two 32-bit x 32-word FIFO buffers incorporated each for reception and transmission

The following figure shows the main components of PCMn.

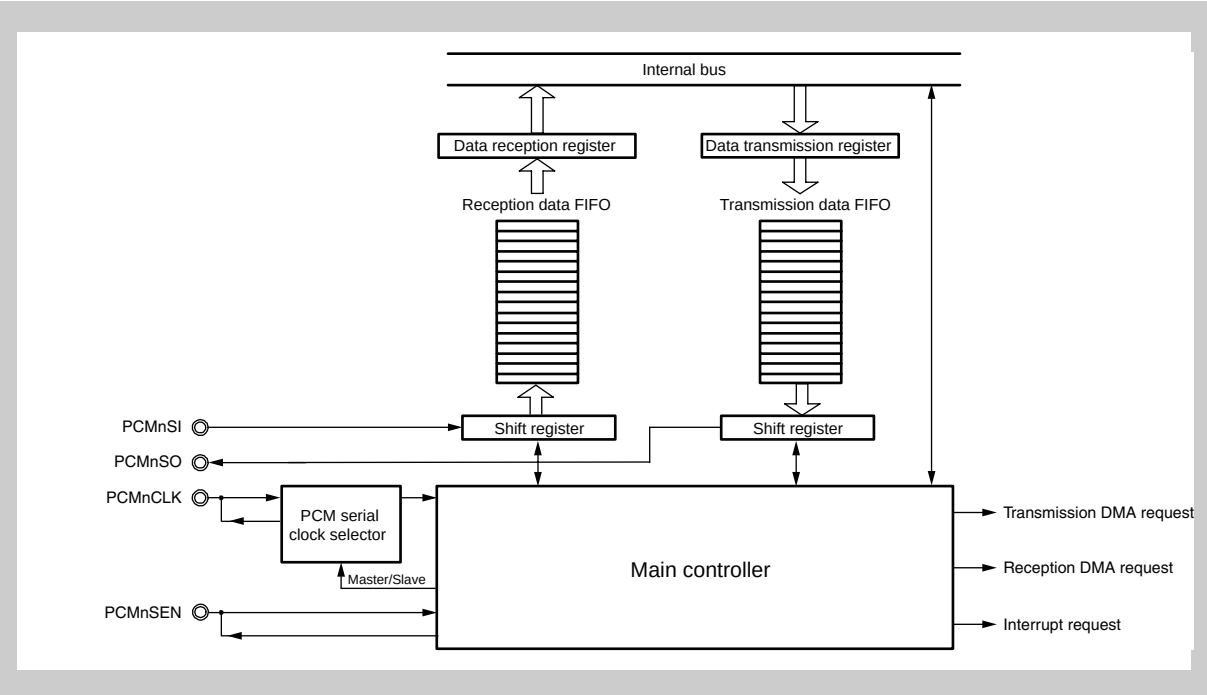


Figure 28-4 Block diagram of PCMn

28.3 Functional Description

The PCM interface is a serial interface used for connecting an external voice codec device or audio codec device. The PCM interface can operate in one of two modes: slave mode, in which the serial clock and frame synchronization signal are externally received; and master mode, in which the serial clock and frame synchronization signal are generated internally.

The PCM interface supports seven serial interface communication modes.

The PCM interface has a FIFO buffer for transmission/reception and an interface (DMA request issuance function) for transferring data to and from the DMA controller.

28.3.1 Master mode/slave mode setting

Master mode or slave mode can be selected for the serial clock (PCMnCLK) and frame synchronization signal (PCMnSEN).

The PCMnMOD.M_S[1:0] bits are used to select the master or slave mode.

In master mode, the serial clock and frame synchronization signal are generated and output.

In slave mode, the serial clock and frame synchronization signal are received from an external device.

In the initial state after reset release, neither the master mode nor the slave mode is selected.

When the PCM interface is used, one of the modes needs to be selected by using the PCMnMOD.M_S[1:0] bits.

28.3.2 Serial interface timing

One of seven serial interface modes (modes 0 to 6) is selected to perform serial data transmission/reception at the timing corresponding to the selected mode.

The PCMnMOD.MODE_SEL[2:0] bits are used to select the serial interface mode. The frame synchronization signal period, the number of valid reception bits, and the number of valid transmission bits must also be specified.

- Frame synchronization signal period setting:

  Specify the frame period by using the PCMnCYC1.CYC_VAL[7:0] bits.
  The value of the frame synchronization signal period is (CYC_VAL[7:0] bit value + 1).

- Number of valid reception bits (for modes 0 to 4):

  Specify the number of valid reception bits by using the PCMnCYC1.SIB[4:0] bits.
  The number of valid reception bits is (SIB[4:0] bit value + 1).
  A value ranging from 8 to 32 can be specified as the number of valid reception bits.
  Specify these bits so that (PCMnCYC1.CYC_VAL[7:0] bit value + 1) are greater than or equal to (SIB[4:0] bit value + 1).

- Number of valid reception bits (for modes 5 and 6):

  Specify the number of valid reception bits by using the PCMnCYC1.SIB[4:0] bits for phase 1, or the PCMnCYC2.SIB2[4:0] bits for phase 2.
  The number of valid reception bits is ({SIB[4:0] + SIB2[4:0]} bit value + 1).
  A value ranging from 8 to 32 can be specified as the number of valid reception bits.
  The same value must be specified for the SOB[4:0] and SIB[4:0] bits, and the same value must be specified for the SOB2[4:0] and SIB2[4:0] bits (SOB[4:0] = SIB[4:0], SOB2[4:0] = SIB2[4:0]).

- Number of valid transmission bits (for modes 0 to 4):

  Specify the number of valid transmission bits by using the PCMnCYC1.SOB[4:0] bits.
  The number of valid transmission bits is (SOB[4:0] bit value + 1).
  A value ranging from 8 to 32 can be specified as the number of valid transmission bits.
  Specify these bits so that (PCMnCYC1.CYC_VAL[7:0] bit value + 1) are greater than or equal to (SOB[4:0] bit value + 1).

- Number of valid transmission bits (for modes 5 and 6):

  Specify the number of valid transmission bits by using the PCMnCYC1.SOB[4:0] bits for phase 1, or the PCMnCYC2.SOB2[4:0] bits for phase 2.
  The number of valid transmission bits is ({SOB[4:0] + SOB2[4:0]} bit value + 1).
  A value ranging from 8 to 32 can be specified as the number of valid transmission bits.
  The same value must be specified for the SOB[4:0] and SIB[4:0] bits, and the same value must be specified for the SOB2[4:0] and SIB2[4:0] bits (SOB[4:0] = SIB[4:0], SOB2[4:0] = SIB2[4:0]).

- Serial I/O data

  In all modes, serial I/O data is transferred MSB-first.
  Use the PCMnCFG.PCMnCFGC bit to specify the timing at which signals are output and sampled in each mode.
  The PCMnCFG.PCMnCFGS bit can also be used to specify the PCMnSEN pin active level.
  The timing diagrams shown on the following pages present typical examples in which the PCMnCFG.PCMnCFGC bit is set to 1, and the PCMnCFGS bit is set to 1. Note that, if the PCMnCFGC or PCMnCFGS bit is cleared to 0, the waveforms output from the PCMnCLK and PCMnSEN pins are inverted.

(1) Mode 0 serial interface timing

Figure 28-5 "Serial interface timing in mode 0" shows the serial interface timing in mode 0.

(a-1) Frame synchronization signal (PCMnSEN) output timing in master mode

The frame synchronization signal (PCMnSEN) is output at the rising edge of the serial clock signal (PCMnCLK).

(a-2) Frame synchronization signal (PCMnSEN) sampling timing in slave mode

The frame synchronization signal (PCMnSEN) is sampled at the falling edge of the serial clock signal (PCMnCLK).

(b) Transmission data (PCMnSO) output timing

Transmission data (PCMnSO) is output at the rising edge of the serial clock signal (PCMnCLK).

(c) Reception data (PCMnSI) sampling timing

Reception data (PCMnSI) is sampled at the falling edge of the serial clock signal (PCMnCLK).

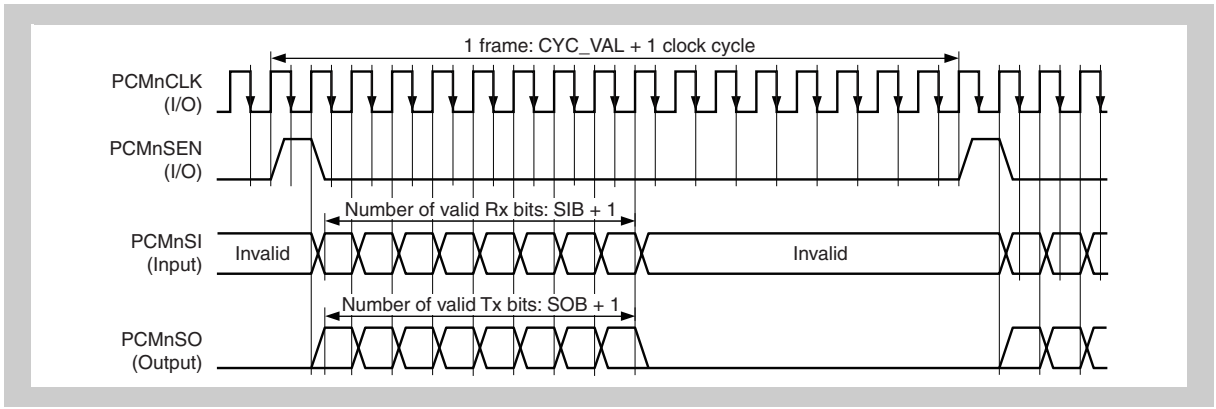


Figure 28-5 Serial interface timing in mode 0

CYC_VAL, SIB, and SOB in the above figure refer to the following:

- CYC_VAL: PCMnCYC1.CYC_VAL[7:0] bit value
- SIB: PCMnCYC1.SIB[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value

(2) Mode 1 serial interface timing

Figure 28-6 "Serial interface timing in mode 1" shows the serial interface timing in mode 1.

(a-1) Frame synchronization signal (PCMnSEN) output timing in master mode

The frame synchronization signal (PCMnSEN) is output at the falling edge of the serial clock signal (PCMnCLK).

(a-2) Frame synchronization signal (PCMnSEN) sampling timing in slave mode

The frame synchronization signal (PCMnSEN) is sampled at the rising edge of the serial clock signal (PCMnCLK).

(b) Transmission data (PCMnSO) output timing

Transmission data (PCMnSO) is output at the rising edge of the serial clock signal (PCMnCLK).

(c) Reception data (PCMnSI) sampling timing

Reception data (PCMnSI) is sampled at the rising edge of the serial clock signal (PCMnCLK).

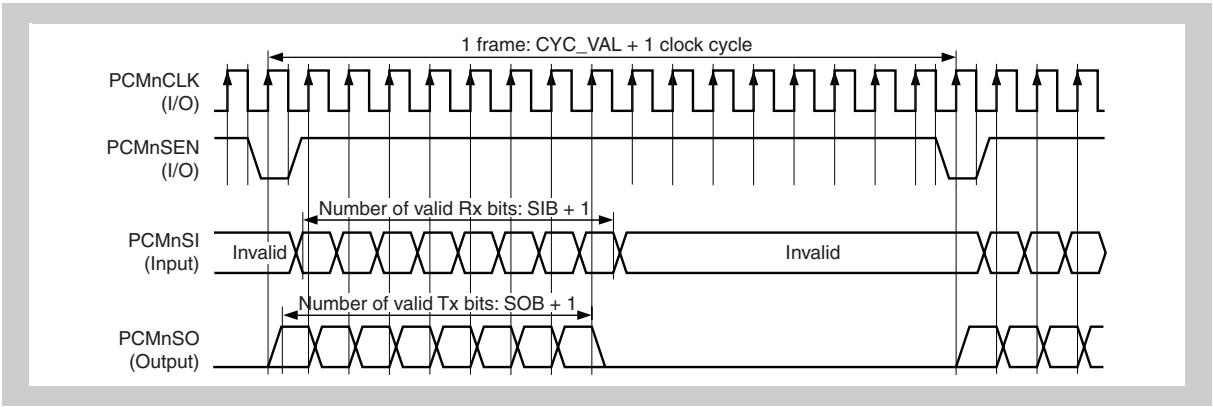


Figure 28-6 Serial interface timing in mode 1

CYC_VAL, SIB, and SOB in the above figure refer to the following:

- CYC_VAL: PCMnCYC1.CYC_VAL[7:0] bit value
- SIB: PCMnCYC1.SIB[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value

(3) Mode 2 serial interface timing

Figure 28-7 "Serial interface timing in mode 2" shows the serial interface timing in mode 2.

The L-ch data is transferred while the PCMnSEN signal is at low level, and the R-ch data is transferred while the PCMnSEN signal is at high level.

(a-1) Frame synchronization signal (PCMnSEN) output timing in master mode

The frame synchronization signal (PCMnSEN) is output at the falling edge of the serial clock signal (PCMnCLK).

(a-2) Frame synchronization signal (PCMnSEN) sampling timing in slave mode

The frame synchronization signal (PCMnSEN) is sampled at the rising edge of the serial clock signal (PCMnCLK).

(b) Transmission data (PCMnSO) output timing

Transmission data (PCMnSO) is output at the falling edge of the serial clock signal (PCMnCLK).

(c) Reception data (PCMnSI) sampling timing

Reception data (PCMnSI) is sampled at the rising edge of the serial clock signal (PCMnCLK).

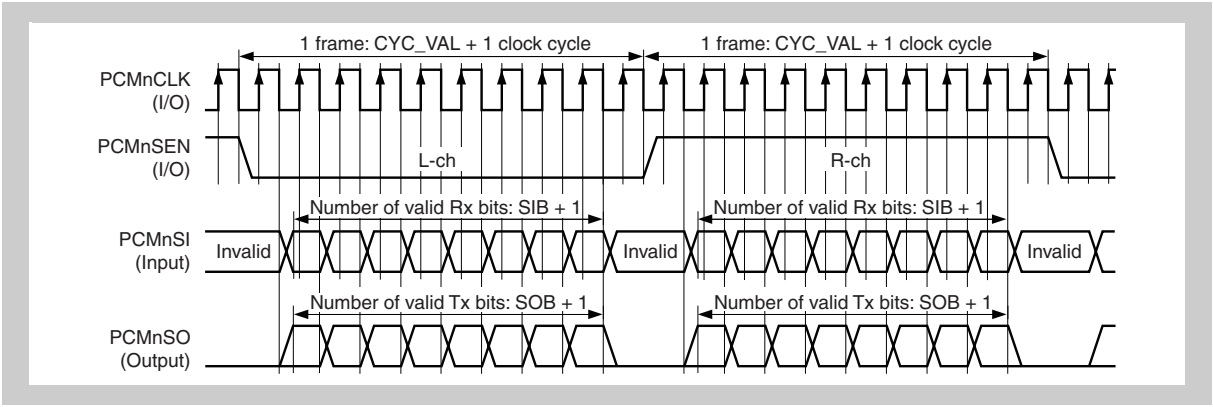


Figure 28-7 Serial interface timing in mode 2

CYC_VAL, SIB, and SOB in the above figure refer to the following:

- CYC_VAL: PCMnCYC1.CYC_VAL[7:0] bit value
- SIB: PCMnCYC1.SIB[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value

(4) Mode 3 serial interface timing

Figure 28-8 "Serial interface timing in mode 3" shows the serial interface timing in mode 3.

The L-ch data is transferred while the PCMnSEN signal is at high level, and the R-ch data is transferred while the PCMnSEN signal is at low level.

(a-1) Frame synchronization signal (PCMnSEN) output timing in master mode

The frame synchronization signal (PCMnSEN) is output at the falling edge of the serial clock signal (PCMnCLK).

(a-2) Frame synchronization signal (PCMnSEN) sampling timing in slave mode

The frame synchronization signal (PCMnSEN) is sampled at the rising edge of the serial clock signal (PCMnCLK).

(b) Transmission data (PCMnSO) output timing

Transmission data (PCMnSO) is output at the falling edge of the serial clock signal (PCMnCLK).

(c) Reception data (PCMnSI) sampling timing

Reception data (PCMnSI) is sampled at the rising edge of the serial clock signal (PCMnCLK).

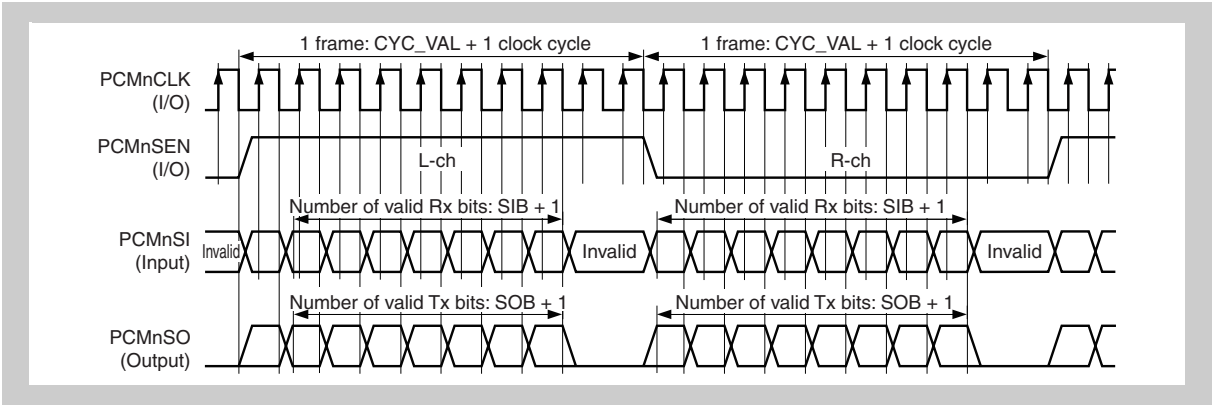


Figure 28-8 Serial interface timing in mode 3

CYC_VAL, SIB, and SOB in the above figure refer to the following:

- CYC_VAL: PCMnCYC1.CYC_VAL[7:0] bit value
- SIB: PCMnCYC1.SIB[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value

(5) Mode 4 serial interface timing

Figure 28-9 "Serial interface timing in mode 4" shows the serial interface timing in mode 4.

The L-ch data is transferred while the PCMnSEN signal is at high level, and the R-ch data is transferred while the PCMnSEN signal is at low level.

(a-1) Frame synchronization signal (PCMnSEN) output timing in master mode

The frame synchronization signal (PCMnSEN) is output at the falling edge of the serial clock signal (PCMnCLK).

(a-2) Frame synchronization signal (PCMnSEN) sampling timing in slave mode

The frame synchronization signal (PCMnSEN) is sampled at the rising edge of the serial clock signal (PCMnCLK).

(b) Transmission data (PCMnSO) output timing

Transmission data (PCMnSO) is output at the falling edge of the serial clock signal (PCMnCLK).

(c) Reception data (PCMnSI) sampling timing

Reception data (PCMnSI) is sampled at the rising edge of the serial clock signal (PCMnCLK).

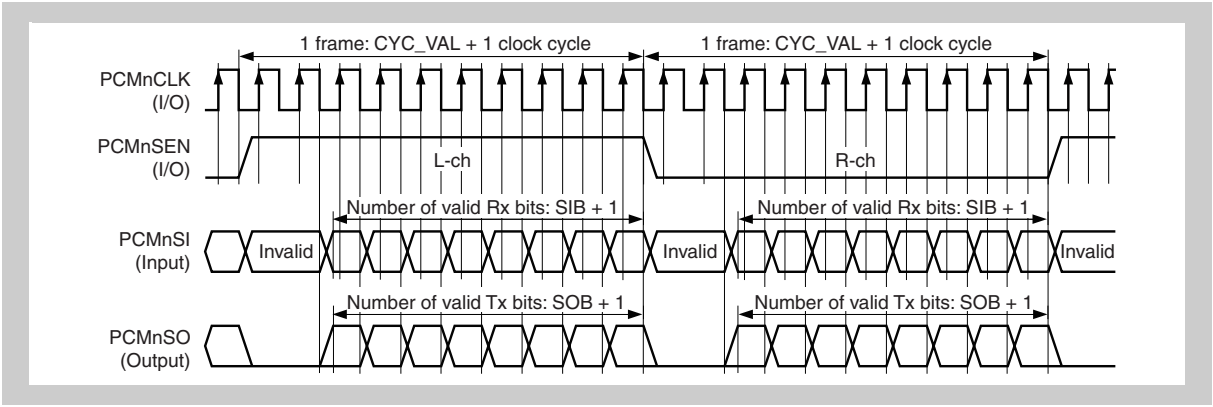


Figure 28-9 Serial interface timing in mode 4

CYC_VAL, SIB, and SOB in the above figure refer to the following:

- CYC_VAL: PCMnCYC1.CYC_VAL[7:0] bit value
- SIB: PCMnCYC1.SIB[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value

(6) Mode 5 serial interface timing

Figure 28-10 "Serial interface timing in mode 5" shows the serial interface timing in mode 5.

When Word_count (number of words) is 0 for phase 1 or 2, a single-phase operation is performed.

It is prohibited to clear both Word_count for phase 1 and Word_count for phase 2 to 0.

For simultaneous transmission/reception, specify the same number of bits/words for transmission/reception in each phase (PCMnCYC1.SIB[4:0] = PCMnCYC1.SOB[4:0], PCMnCYC2.SIB2[4:0] = PCMnCYC2.SOB2[4:0]).

Word_size and Word_count can be specified individually for phase 1 and phase 2. (It is possible to specify values such that SIB[4:0] is not equal to SIB2[4:0], SOB[4:0] is not equal to SOB2[4:0], and PCMnCYC1.CYC_VAL[7:0] is not equal to PCMnCYC2.CYC_VAL2[7:0].)

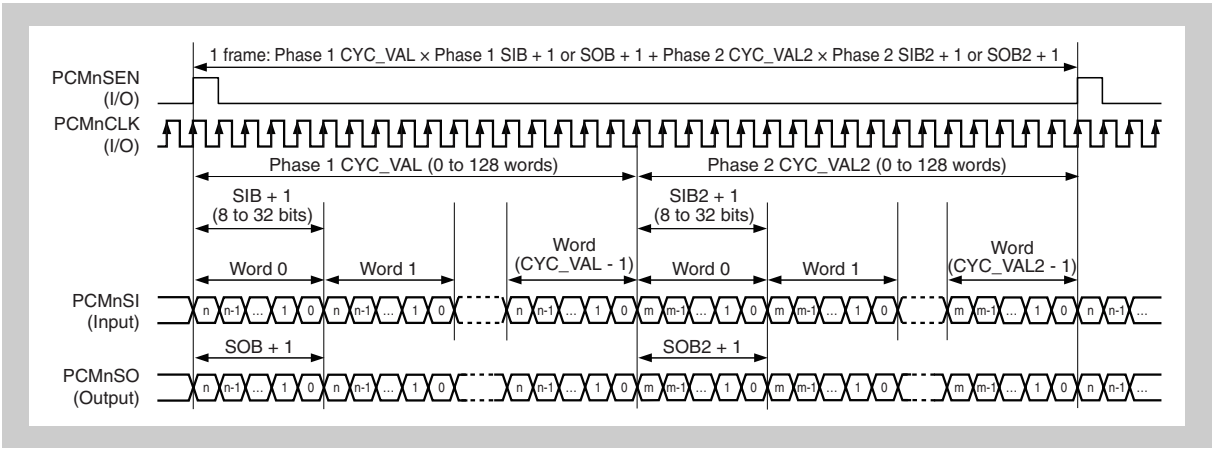


Figure 28-10 Serial interface timing in mode 5

CYC_VAL, CYC_VAL2, SIB, SIB2, SOB, and SOB2 in the above figure refer to the following:

- CYC_VAL: PCMnCYC1.CYC_VAL[7:0] bit value
- CYC_VAL2: PCMnCYC2.CYC_VAL2[7:0] bit value
- SIB: PCMnCYC1.SIB[4:0] bit value
- SIB2: PCMnCYC2.SIB2[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value
- SOB2: PCMnCYC2.SOB2[4:0] bit value

(7) Mode 6 serial interface timing

Figure 28-11 "Serial interface timing in mode 6" shows the serial interface timing in mode 6.

When Word_count (number of words) is 0 for phase 1 or 2, a single-phase operation is performed.

It is prohibited to clear both Word_count for phase 1 and Word_count for phase 2 to 0.

For simultaneous transmission/reception, specify the same number of bits/words for transmission/reception in each phase (PCMnCYC1.SIB[4:0] = PCMnCYC1.SOB[4:0], PCMnCYC2.SIB2[4:0] = PCMnCYC2.SOB2[4:0]).

(It is possible to specify values such that SIB[4:0] is not equal to SIB2[4:0], SOB[4:0] is not equal to SOB2[4:0], and PCMnCYC1.CYC_VAL[7:0] is not equal to PCMnCYC2.CYC_VAL2[7:0].)

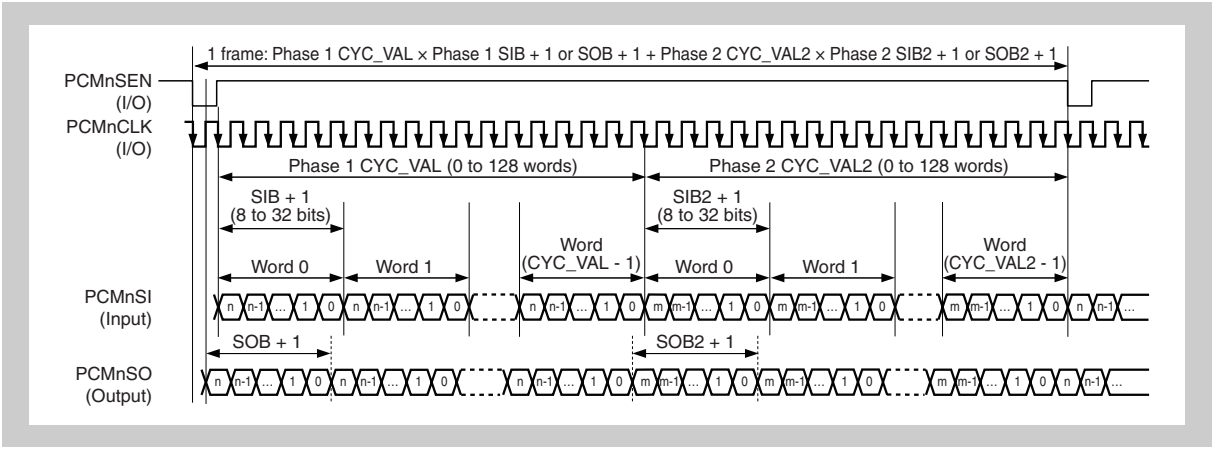


Figure 28-11 Serial interface timing in mode 6

CYC_VAL, CYC_VAL2, SIB, SIB2, SOB, and SOB2 in the above figure refer to the following:

- CYC_VAL: PCMnCYC1.CYC_VAL[7:0] bit value
- CYC_VAL2: PCMnCYC2.CYC_VAL2[7:0] bit value
- SIB: PCMnCYC1.SIB[4:0] bit value
- SIB2: PCMnCYC2.SIB2[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value
- SOB2: PCMnCYC2.SOB2[4:0] bit value

28.3.3 Data padding

Data padding can be performed when the data bit length is 8 or 16 bits.

This refers to the operation in which four 8-bit units or two 16-bit units are padded to one word (32 bits), because the FIFO bit width is 32 bits.

Data padding can be enabled or disabled by using the PCMnCYC1 register.

Data padding is prohibited in mode 5 or 6.

(1) Data padding on transmission side

(a) Without data padding

Example Transmission data bit length = 8 bits (PCMnCYC1.SOB[4:0] bits = 07H)

Data padding on the transmission side is disabled (PCMnCYC1.TX_PD bit = 0)

Writing xxxx_xxF1H by using the PCMnTXQ register

Writing xxxx_xx23H by using the PCMnTXQ register

Writing xxxx_xx45H by using the PCMnTXQ register

Writing xxxx_xx67H by using the PCMnTXQ register

Figure 28-12 Data padding on transmission side: Without data padding

Data with a length other than the transmission data bit length is ignored (x: Don't care).

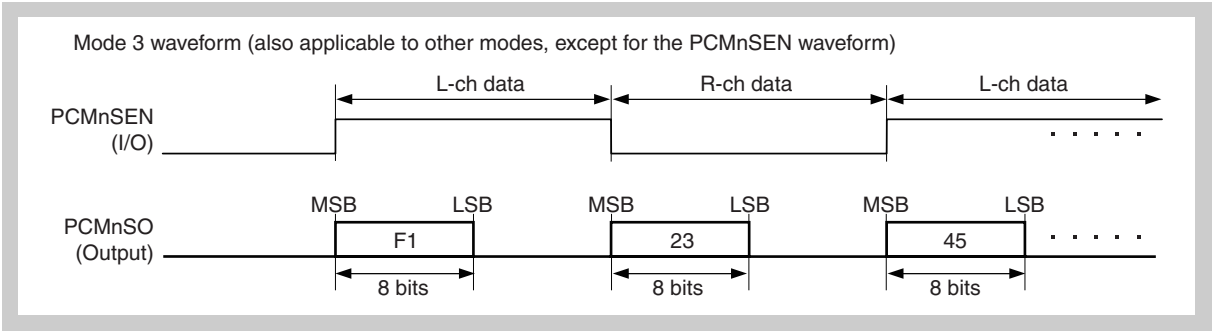


Figure 28-13 Transmission waveform in mode 3

(b) With data padding

Example Transmission data bit length = 8 bits (PCMnCYC1.SOB[4:0] bits = 07H)

Data padding on the transmission side is enabled (PCMnCYC1.TX_PD bit = 1)

Writing F123_4567H by using the PCMnTXQ register

Writing 89AB_CDEFH by using the PCMnTXQ register

Writing 2468_ACE0H by using the PCMnTXQ register

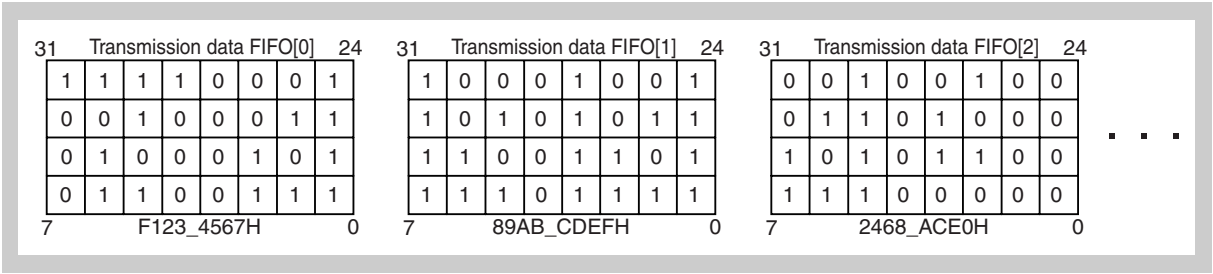


Figure 28-14 Data padding on transmission side: With data padding

Note The signal waveform is the same as that in *Figure 28-13 “Transmission waveform in mode 3”*.

(2) Data padding on reception side

(a) Without data padding

Example Reception data bit length = 16 bits (PCMnCYC1.SIB[4:0] bits = 0FH)

Data padding on the reception side is disabled (PCMnCYC1.RX_PD bit = 0)

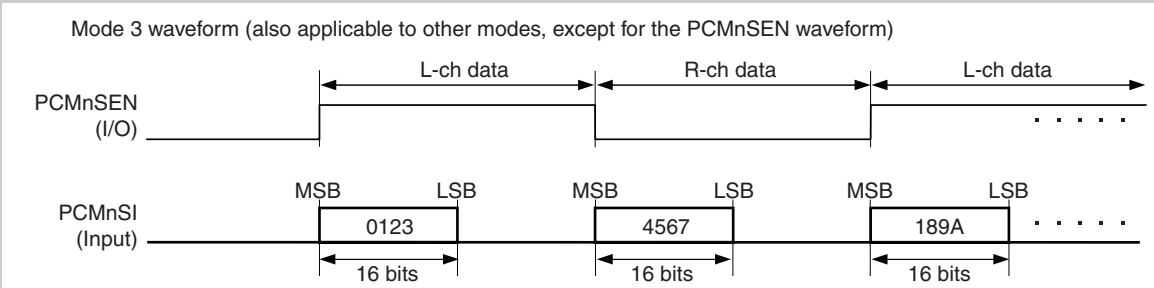


Figure 28-15 Transmission waveform in mode 3

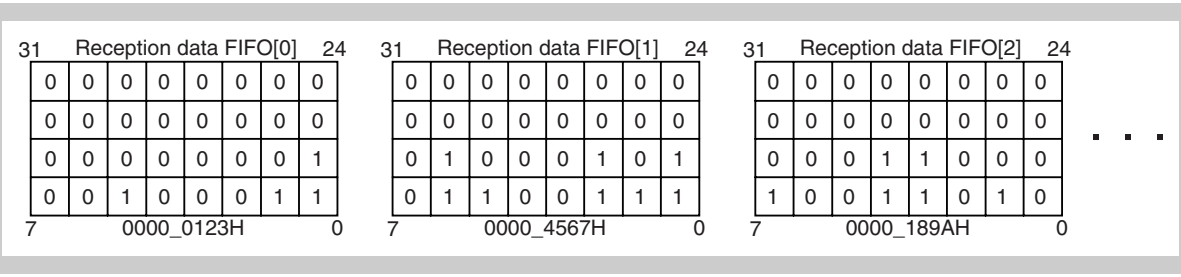


Figure 28-16 Data padding on reception side: Without data padding

Data exceeding the reception data bit length is padded with 0.

Reading data (0000_0123H) from reception data FIFO [0] by using the PCMnRXQ register

Reading data (0000_4567H) from reception data FIFO [1] by using the PCMnRXQ register

Reading data (0000_189AH) from reception data FIFO [2] by using the PCMnRXQ register

(b) With data padding

Example Reception data bit length = 16 bits (PCMnCYC1.SIB[4:0] bits = 0FH)

Data padding on the reception side is enabled (PCMnCYC1.RX_PD bit = 1)

Note The signal waveform is the same as that in *Figure 28-15 "Transmission waveform in mode 3"*.

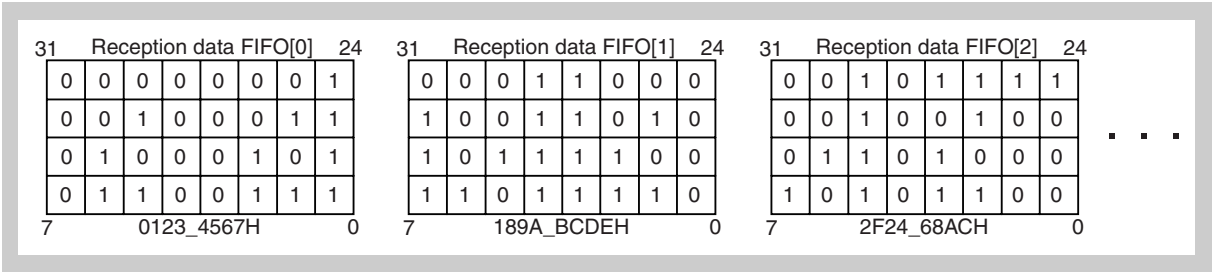


Figure 28-17 Data padding on reception side: With data padding

Reading data (0123_4567H) from reception data FIFO [0] by using the PCMnRXQ register

Reading data (189A_BCDEH) from reception data FIFO [1] by using the PCMnRXQ register

Reading data (2F24_68ACH) from reception data FIFO [2] by using the PCMnRXQ register

Caution If 32 bits of data from the PCMnSI pin does not accumulate in the reception data FIFO, the data is not considered valid and the last data cannot be read.

28.3.4 FIFO operation

(1) In modes 0 and 1

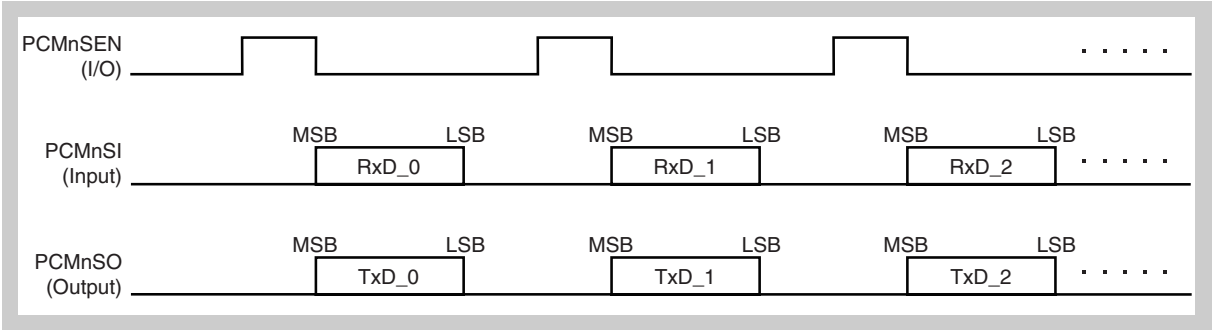


Figure 28-18 FIFO mode operation in modes 0 and 1

(a) Transmission data FIFO

1. Data is stored in the transmission data FIFO in the order that the data was written to the PCMnTXQ register from the host.
2. Data equivalent to one word is taken out of the transmission data FIFO according to the transmission data FIFO counter.

   Based on the value n specified for the PCMnCYC1.SOB[4:0] bits, serial data is output from the PCMnSO pin in order from bit n (MSB) to bit 0 (LSB).
3. After the data of bit 0 (LSB) is transmitted, the transmission data FIFO counter is incremented, and the same processing as *2* is repeated when transmission of the next frame starts. The counter range is 0 to 31 (1FH) and the counter returns to 0 when counting up from 31.

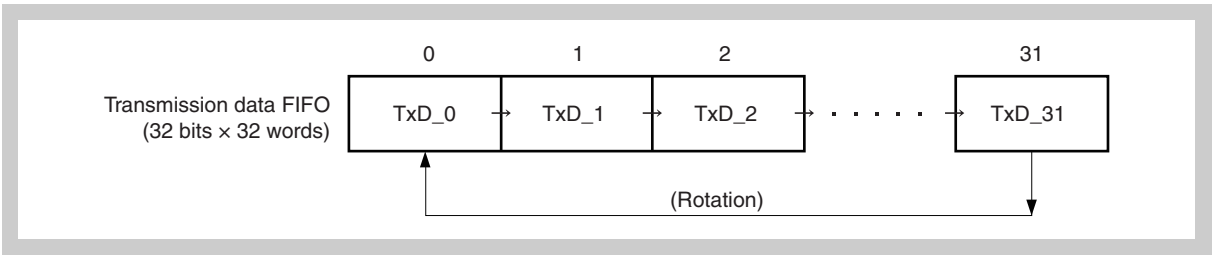


Figure 28-19 Transmission data FIFO in modes 0 and 1

(b) Reception data FIFO

1. The reception data FIFO receives the serial data from the PCMnSI pin.
2. Based on the value m specified for the PCMnCYC1.SIB[4:0] bits, the received bits are stored in the reception data FIFO word by word in order from bit m (MSB) to bit 0 (LSB). The bits higher than bit m are padded with 0.
3. After the data of bit 0 (LSB) is stored, the reception data FIFO counter is incremented, and the same processing as *2* is performed when the next frame is received. The counter range is 0 to 31 (1FH) and the counter returns to 0 when counting up from 31.

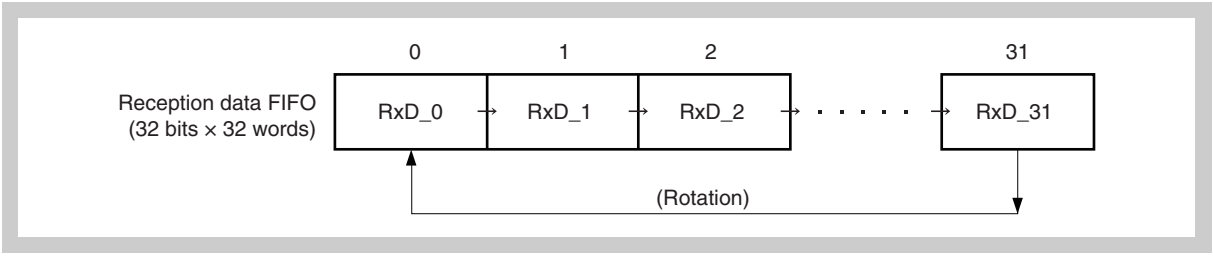


Figure 28-20 Reception data FIFO in modes 0 and 1

(2) In modes 2 to 4

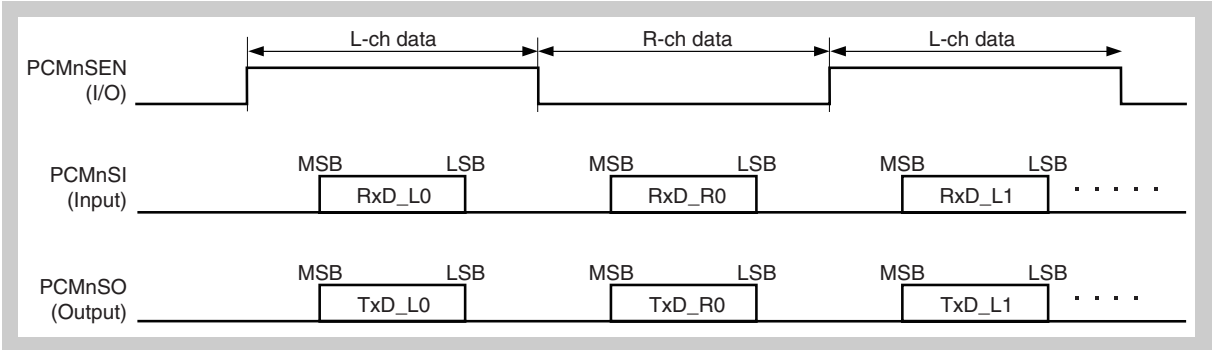


Figure 28-21 FIFO mode operation in modes 2 to 4

In modes 2 to 4, L-ch data and R-ch data are alternately stored in the reception data FIFO and transmission data FIFO because the L-ch data and the R-ch data are alternately transmitted and received.

Therefore, the L-ch data and the R-ch data need to be alternately written for transmission. For reception, the L-ch data is read first, then the R-ch data.

In all other cases, the FIFO operates the same as in modes 0 and 1.

(a) Transmission data FIFO

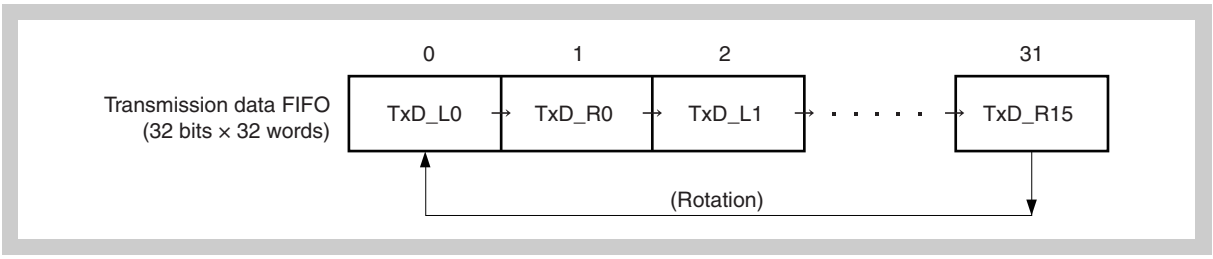


Figure 28-22 Transmission data FIFO in modes 2 to 4

(b) Reception data FIFO

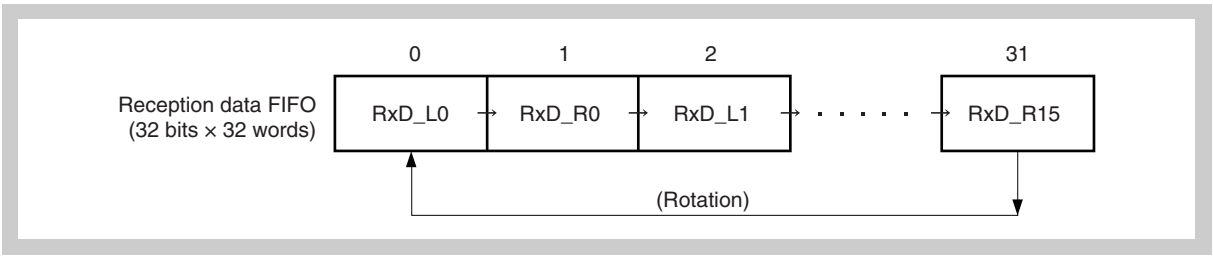


Figure 28-23 Reception data FIFO in modes 2 to 4

Note To store padded data in the FIFO, see *28.3.3 "Data padding"*.

(3) In modes 5 and 6

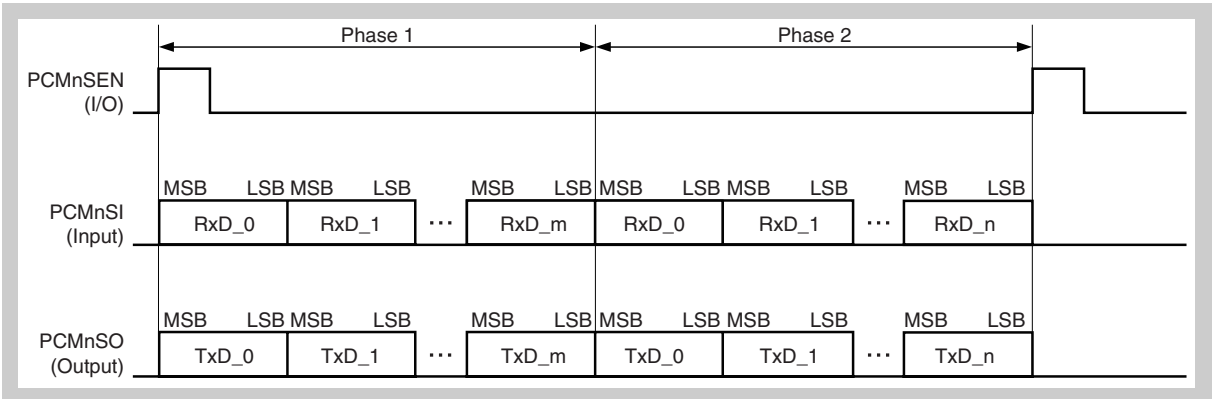


Figure 28-24 FIFO mode operation in modes 5 and 6

(a) Transmission data FIFO

1. Data is stored in the transmission data FIFO in the order that the data was written to the PCMnTXQ register from the host.
2. Data equivalent to one word is taken out of the transmission data FIFO according to the transmission data FIFO counter.

   Based on the values n and m specified for the PCMnCYC1.SOB[4:0] and PCMnCYC2.SOB2[4:0] bits, serial data is output from the PCMnSO pin as the phase 1 transmission data equivalent to the number of words specified for the PCMnCYC1.CYC_VAL[7:0] bits in order from bit m (MSB) to bit 0 (LSB), and subsequently serial data transmitted in phase 2 equivalent to the number of words specified for the PCMnCYC2.CYC_VAL2[7:0] bits is output from the PCMnSO pin in order from bit n (MSB) to bit 0 (LSB).
3. After one word of data up to bit 0 (LSB) is transmitted, the transmission data FIFO counter is incremented, and transmission of the next data word starts. After the data up to bit 0 (LSB) is transmitted, the transmission data FIFO counter is incremented again, and the same processing is repeated. The transmission data FIFO counter range is 0 to 31 (1FH) and the counter returns to 0 when counting up from 31.

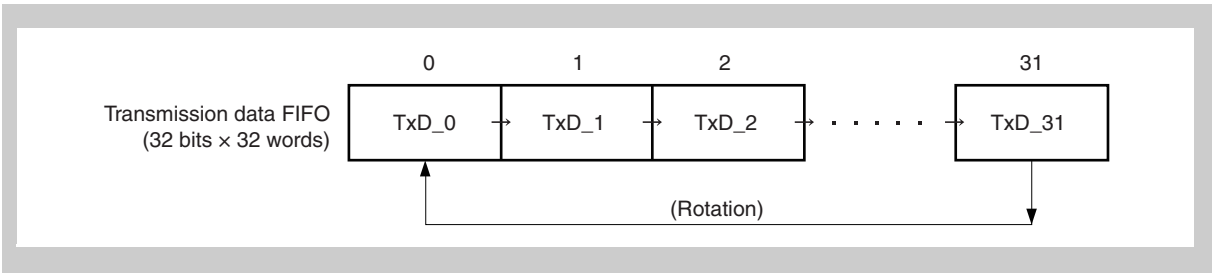


Figure 28-25 Transmission data FIFO in modes 5 and 6

(b) Reception data FIFO

1. The reception data FIFO receives the serial data from the PCMnSI pin.
2. Based on the values n and m specified for the PCMnCYC1.SIB[4:0] and PCMnCYC2.SIB2[4:0] bits, serial data received in phase 1 is stored in the reception data FIFO up to the number of words specified for the PCMnCYC1.CYC_VAL[7:0] bits in order from bit m (MSB) to bit 0 (LSB), and subsequently serial data received in phase 2 equivalent to the number of words specified for the PCMnCYC2.CYC_VAL2[7:0] bits is stored in the reception data FIFO in order from bit n (MSB) to bit 0 (LSB). Bits higher than m and n are padded with 0 if the received data is less than 32 bits.
3. After one word of data up to bit 0 (LSB) is received, the reception data FIFO counter is incremented, and reception of the next data word starts. After the data is received up to bit 0 (LSB), the reception data FIFO counter is incremented again, and the same processing is repeated. The reception data FIFO counter range is 0 to 31 (1FH) and the counter returns to 0 when counting up from 31.

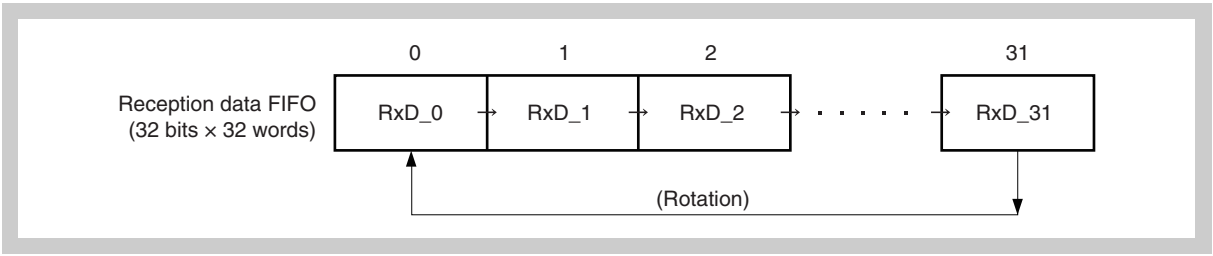


Figure 28-26 Reception data FIFO in modes 5 and 6

28.3.5 Data transfer status (status/error/interrupt source)

Table 28-8 "Data transfer statuses (status/error/interrupt source)" lists data transfer statuses.

The data transfer status can be checked by reading the PCMnRAW register. When an error is detected, the transmission or reception that was being executed when the error occurred stops. If the operation is being performed in master mode, however, the serial clock signal (PCMnCLK) and frame synchronization signal (PCMnSEN) continue to be output.

Table 28-8 Data transfer statuses (status/error/interrupt source)

Status	Status/error details and operation
RX_RENRAW	Indicates that reading the received data is enabled. This flag and the DMA reception request signal (PCMnRDMARQ) are simultaneously set or cleared. 0: There is no unread data in the reception data FIFO, or reception is disabled: PCMnTEC.RX_ENCLR bit = 1. 1: There is valid data of at least one word in the reception data FIFO (that is, there is data that has not been read by the DMAC).
RX_ORERAW	Indicates that an overrun error occurred at the reception data FIFO. 0: Reception is disabled: PCMnTEC.RX_ENCLR bit = 1 1: An attempt was made to store new data in the reception data FIFO before the existing data was read by the DMAC. (An attempt was made to write the serial data received from the PCMnSI pin to the reception data FIFO while the reception data FIFO was full.)
RX_URERAW	Indicates that an underrun error occurred at the reception data FIFO. 0: Reception is disabled: PCMnTEC.RX_ENCLR bit = 1 1: There is no valid data in the reception data FIFO. The reception data FIFO was read even though it contained no valid data (source RX_RENRAW = 0) (that is, nonexistent reception data was read).
RX_FRERAW	Indicates that a serial interface synchronization error[a] occurred in the reception controller. 0: Reception was disabled (by setting the PCMnTEC.RX_ENCLR bit to 1) after the source was cleared (by setting the PCMnICL.RX_FRECLR bit to 1). 1: A deviation in the PCMnSEN signal was detected (only in case of slave operation).
TX_WENRAW	Indicates that writing transmission data is enabled.[b] This flag and the DMA transmission request signal (PCMnTDMARQ) are simultaneously set or cleared. 0: The transmission data FIFO is full and writing can no longer be performed, or transmission is disabled: PCMnTEC.TX_ENCLR bit = 1. 1: The transmission data FIFO is not full (the transmission data FIFO has free space) and writing can be performed.
TX_ORERAW	Indicates that an overrun error occurred at the transmission data FIFO. 0: Transmission is disabled: PCMnTEC.TX_ENCLR bit = 1 1: An attempt was made to write data to the transmission data FIFO while the FIFO was full (source TX_WENRAW = 0).
TX_URERAW	Indicates that an underrun error occurred at the transmission data FIFO. 0: Transmission was disabled (by setting the PCMnTEC.TX_ENCLR bit to 1) after the source was cleared (by setting the PCMnICL.TX_URECLR bit to 1). 1: All transmission data has been output from the PCMnSO pin and the transmission data FIFO is empty. (Either the completion of transmission of all data has been reported, or the serial data transmission speed is faster than the speed of writing to the transmission data FIFO.)
TX_FRERAW	Indicates that a serial interface synchronization error[a] occurred in the transmission controller. 0: Transmission was disabled (by setting the PCMnTEC.TX_ENCLR bit to 1) after the source was cleared (by setting the PCMnICL.TX_FRECLR bit to 1). 1: A deviation in the PCMnSEN signal was detected (only in case of slave operation).

a) A synchronization error occurs when the PCMnSEN signal does not change for the number of clock cycles per frame specified for the PCMnCYC1.CYC_VAL[7:0], SIB[4:0], and SOB[4:0] bits (for modes 0 to 4), or the PCMnCYC2.CYC_VAL2[7:0], SIB2[4:0], and SOB2[4:0] bits (for modes 5 and 6) during communication (the change depends on the format of modes 0 to 6). A synchronization error also occurs when the transmission block and the reception block fall out of synchronization.

b) This flag is cleared during a reset period and set when 1PCLK is input after the reset period ends.

Note RX_FRERAW or TX_FRERAW is set when a synchronization error is detected in the reception block or transmission block. When both reception and transmission are enabled, the flag of the block in which the error occurred first is set (although this error actually occurs almost simultaneously in both blocks).

28.3.6 Error recovery

(1) Recovery from reception errors

When a reception error (RX_ORE, RX_URE, or RX_FRE) is detected, reception stops.

When a reception error is detected, be sure to disable reception and wait for error recovery before resuming reception.

When a reception error is detected, perform recovery processing according to the procedure below.

1. Clear the reception enable bit (set the PCMnTEC.RX_ENCLR bit to 1) to disable reception.
2. Wait for all reception error flags (RX_ORE, RX_URE, and RX_FRE) to be cleared to 0 to confirm error recovery.
3. Set the reception enable bit (set the PCMnTEN.RX_EN bit to 1) to resume reception.

(2) Recovery from transmission errors

When a transmission error (TX_ORE, TX_URE, or TX_FRE) is detected, transmission stops.

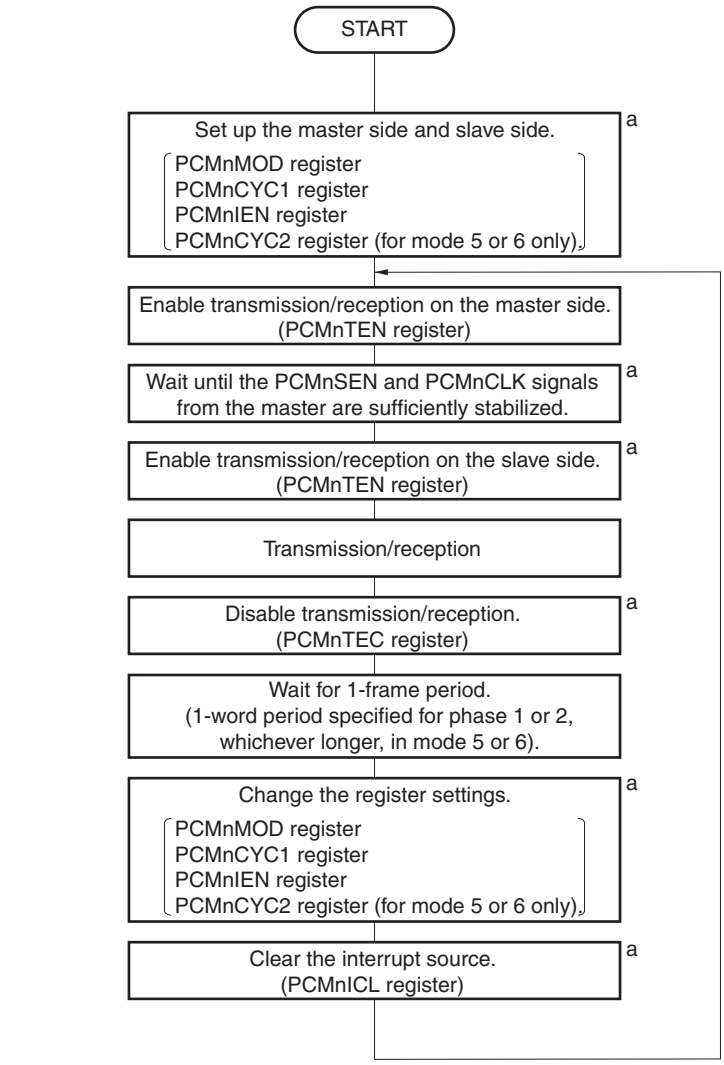
When a transmission error is detected, be sure to disable transmission and wait for error recovery before resuming transmission.

When a transmission error is detected, perform recovery processing according to the procedure below.

1. Clear the transmission enable bit (set the PCMnTEC.TX_ENCLR bit to 1) to disable transmission.
2. Wait for all transmission error flags (TX_ORE, TX_URE, and TX_FRE) to be cleared to 0 to confirm error recovery.
3. Set the transmission enable bit (set the PCMnTEN.TX_EN bit to 1) to resume transmission.

28.4 Setting Sequence

The following figure shows the PCM register setting procedure.



a) In single unit operation, configuring the slave side is not required.

Figure 28-27 Register setting flow

28.4.1 Example of settings to start communication

(1) Set the operation mode. (PCMnMOD register)

- MODE_SEL[2:0] bits = 011:Mode 3
- M_S[1:0] bits = 10: Slave mode
- Tx_TIM[1:0] bits = 00: Transmission starts when 30 words have been stored in the transmission data FIFO.

(2) Specify the number of clock cycles and transmission/reception data bit length. (PCMnCYC1 register)

- CYC_VAL[7:0] bits = 0001_1111: The number of clock cycles per frame is 32.
- SIB[4:0] bits = 1_0111: The PCMnSI data bit length is 24 bits.
- RX_PD bit = 0: There is no data padding on the receiving side.
- SOB[4:0] bits = 1_0111: The PCMnSO data bit length is 24 bits.
- TX_PD bit = 0: There is no data padding on the transmitting side.

(3) Unmask the interrupt source.

- RM_REN_EN bit = 0: Masks the reception data read enable interrupt source.
- RM_ORE_EN bit = 1: Unmasks the reception overrun error interrupt source.
- RM_URE_EN bit = 1: Unmasks the reception underrun error interrupt source.
- RM_FRE_EN bit = 1: Unmasks the reception synchronization error interrupt source.
- TM_WEN_EN bit = 0: Masks the transmission data write enable interrupt source.
- TM_ORE_EN bit = 1: Unmasks the transmission overrun error interrupt source.
- TM_URE_EN bit = 1: Unmasks the transmission underrun error interrupt source.
- TM_FRE_EN bit = 1: Unmasks the transmission synchronization error interrupt source.

(4) Enable transmission and reception (PCMnTEN register)

- TX_EN bit = 1: Enables transmission.
- RX_EN bit = 1: Enables reception.

When operating the microcontroller as a slave, enable transmission and reception after the PCMnSEN and PCMnCLK signals from the master are sufficiently stabilized.

28.4.2 Examples of serial interface operation in modes 5 and 6

(1) Mode 5 operation examples

(a) Double-phase operation

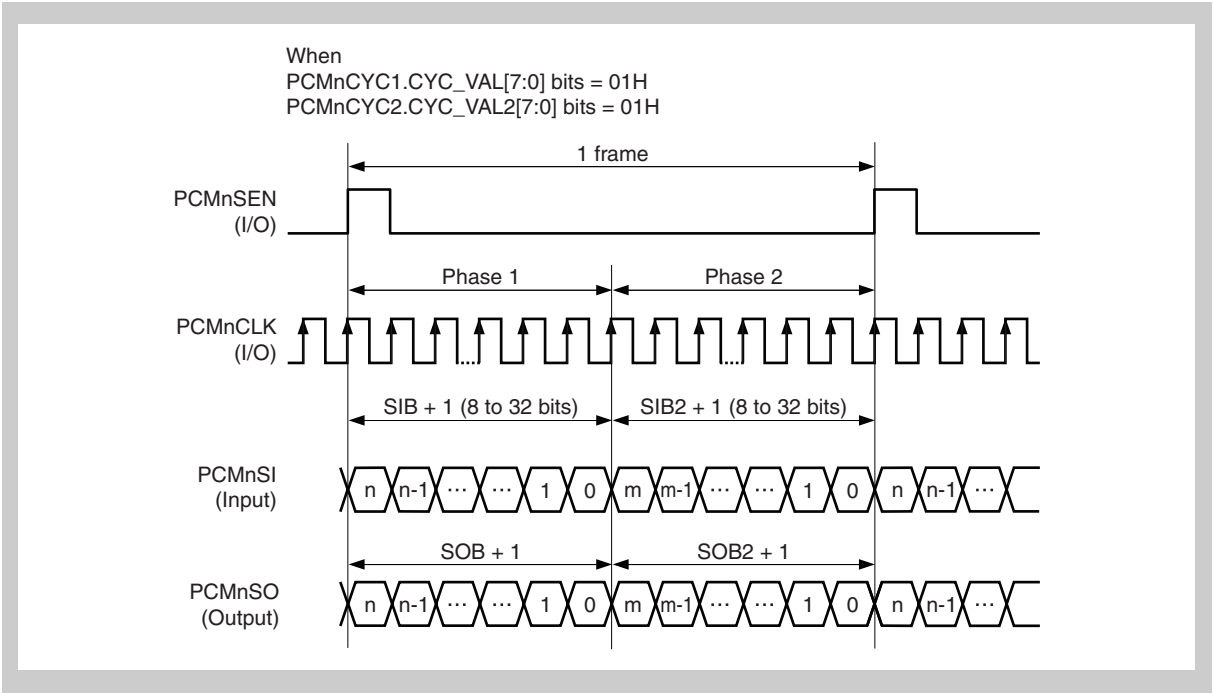


Figure 28-28 Example of double-phase operation in mode 5

SIB, SIB2, SOB, and SOB2 in the above figure refer to the following:

- SIB: PCMnCYC1.SIB[4:0] bit value
- SIB2: PCMnCYC2.SIB2[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value
- SOB2: PCMnCYC2.SOB2[4:0] bit value

(b) Single-phase operation

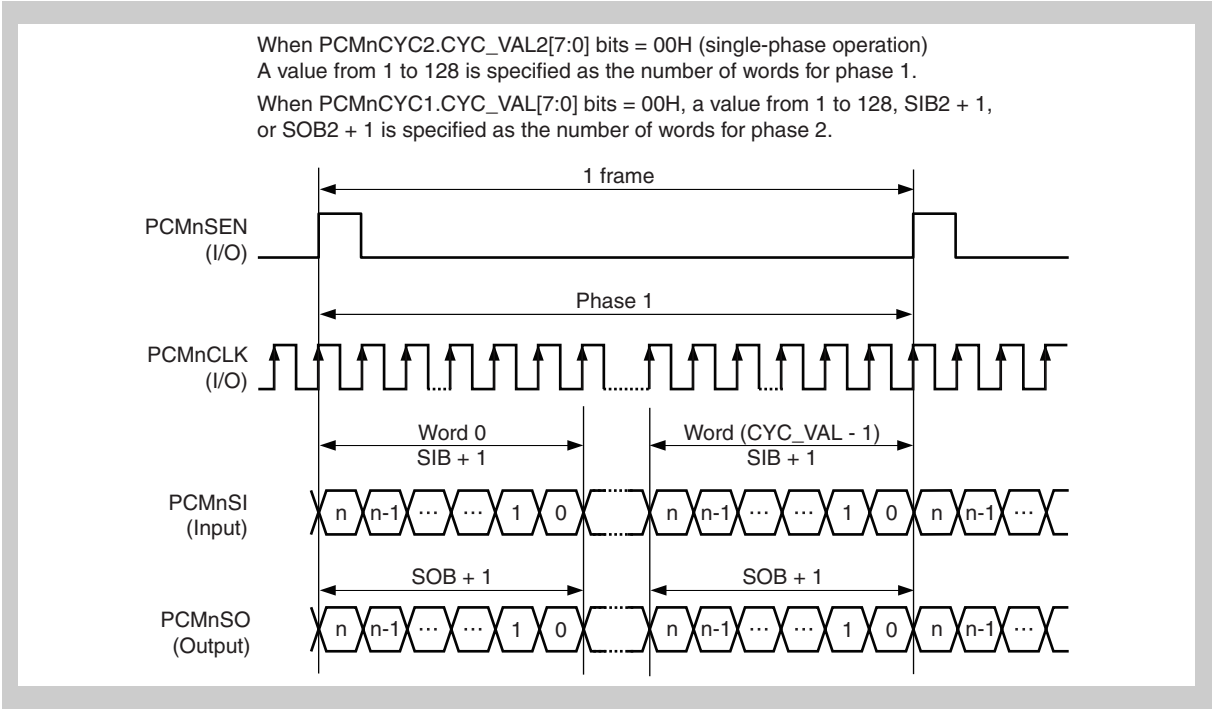


Figure 28-29 Example of single-phase operation in mode 5

SIB and SOB in the above figure refer to the following:

- SIB: PCMnCYC1.SIB[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value

(2) Mode 6 operation examples

(a) Double-phase operation

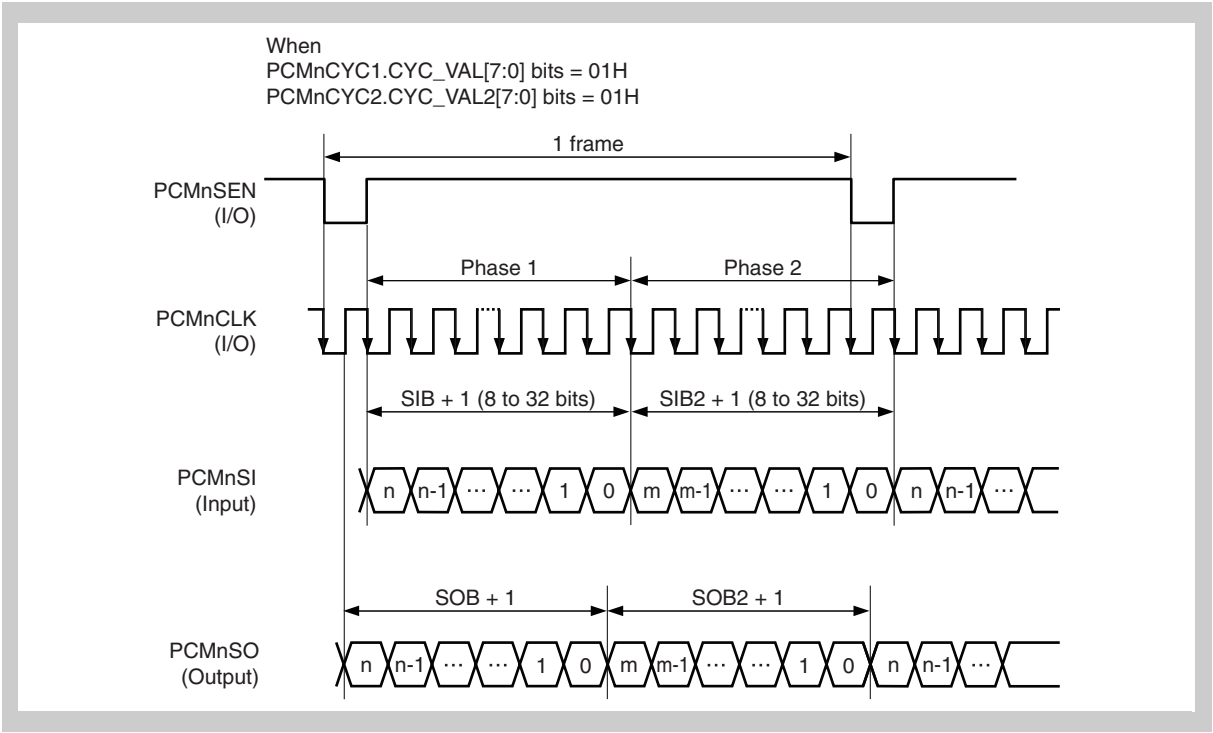


Figure 28-30 Example of double-phase operation in mode 6

SIB, SIB2, SOB, and SOB2 in the above figure refer to the following:

- SIB: PCMnCYC1.SIB[4:0] bit value
- SIB2: PCMnCYC2.SIB2[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value
- SOB2: PCMnCYC2.SOB2[4:0] bit value

(b) Single-phase operation

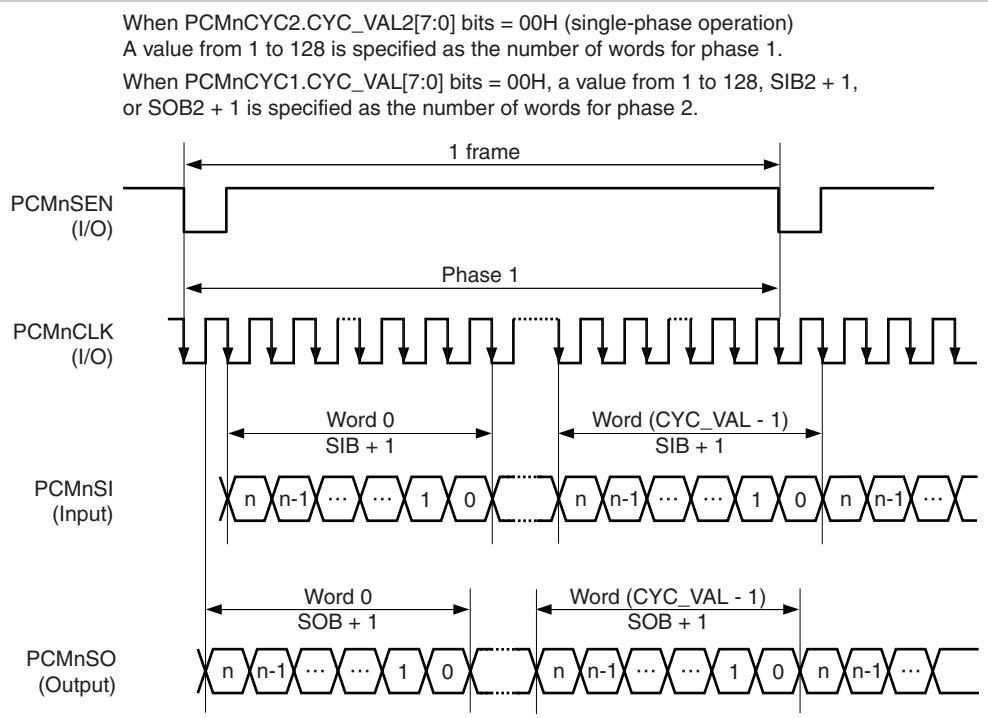


Figure 28-31 Example of single-phase operation in mode 6

SIB and SOB in the above figure refer to the following:

- SIB: PCMnCYC1.SIB[4:0] bit value
- SOB: PCMnCYC1.SOB[4:0] bit value

28.5 Operation

28.5.1 State transitions of transmission block

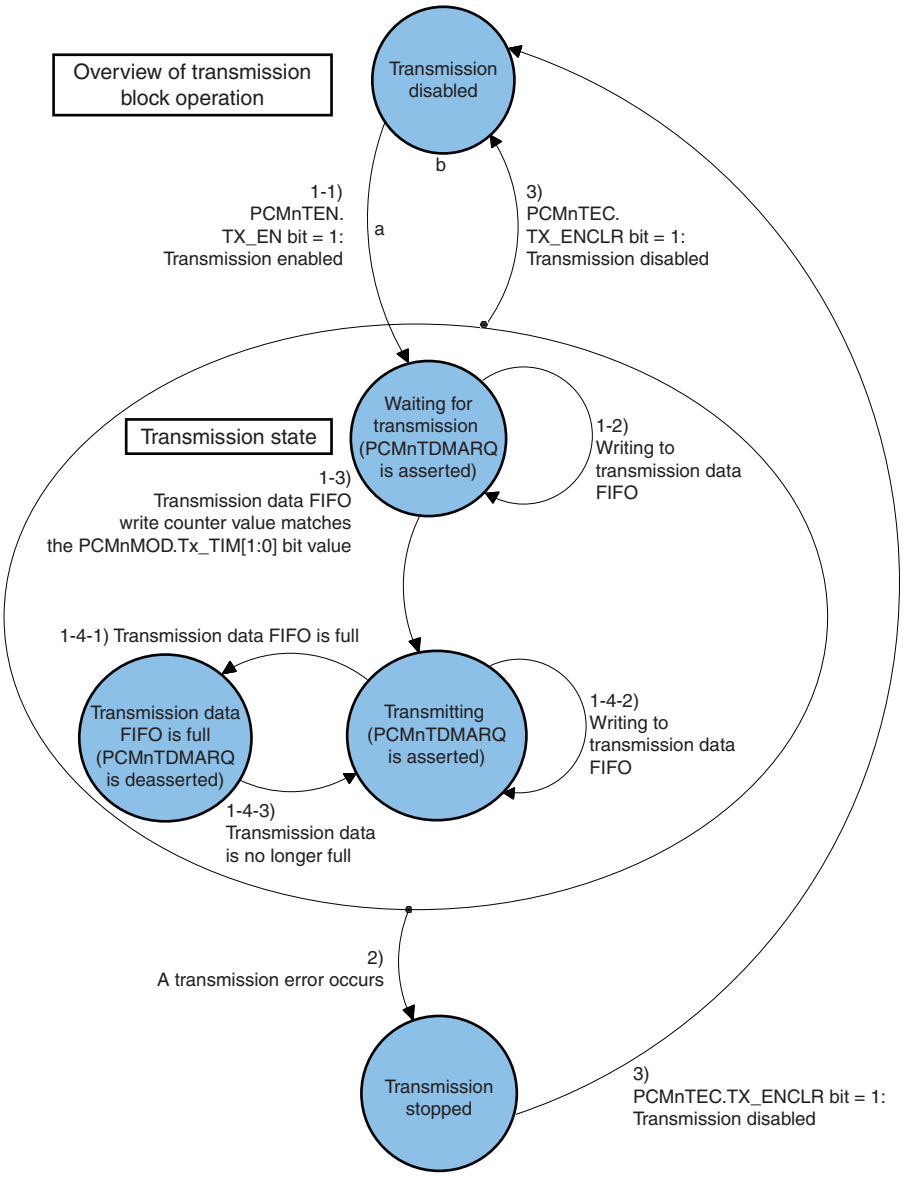


Figure 28-32 State transitions of transmission block

a) To enable transmission again once transmission has been disabled, wait for one frame period (one word period specified for phase 1 or phase 2, whichever is longer, in mode 5 or 6) after transmission is disabled.
Waiting is not required, however, after the operation described in b is performed because waiting is performed in the operation described in *b)*.
When operating the microcontroller as a slave, enable transmission after the PCMnSEN and PCMnCLK signals from the master are sufficiently stabilized.

b) To change settings after transmission stops, wait for one frame period (one word period specified for phase 1 or phase 2, whichever is longer, in mode 5 or 6) after transmission is disabled.

Table 28-9 Statuses during transmission for each operation source

Operation source	Description	Status		
		Transmission disabled	Transmission state	Transmission stopped
PCMnTDMARQ pin	DMA transmission request	Fixed to off	On or off	Fixed to off
PCMnSO pin	Serial data transmission	Fixed to 0	Transmitting	Fixed to 0
PCMnSEN pin during master operation	Serial data synchronization signal	Fixed to 0	Transmitting	Transmitting
TX_WEN flag	Transmission data write enable flag	Cleared to 0	On or off	On/off status retained
TX_ORE flag	Transmission overrun error detected	Cleared to 0	On or off	On/off status retained
TX_URE flag	Transmission underrun error detected	Cleared to 0	On or off	On/off status retained
TX_FRE flag	Transmission synchronization error detected	Cleared to 0	On or off	On or off
TXFIFO	Transmission data FIFO	Write disabled	Write enabled	Write disabled[a]
TX_W_CONT	Write counter of transmission data FIFO	0	Counting	Count value retained
TX_R_CONT	Read counter of transmission data FIFO	0	Counting	Count value retained
TX_WP_NUM[b]	Word number corresponding to FIFO pointed to by write pointer in transmission data FIFO	0	Word number updated	Word number retained
TX_PHASE[b]	Write pointer in transmission data FIFO points to either phase 1 or 2	0	Phase display updated	Phase display retained

a) Writing to the transmission data FIFO in the transmission stopped state might result in a transmission overrun error.

b) Fixed to 0 in modes other than modes 5 and 6.

(1) Description of transmission block state transitions

The item numbers below generally correspond to those in *Figure 28-32* (although there are some that do not).

1) PCMnTEN.TX_EN bit = 1: Transmission state

When operating the microcontroller as a slave, enable transmission after the PCMnSEN and PCMnCLK signals from the master are sufficiently stabilized.

1-1) By setting the transmission enable bit to 1, the state transitions to the transmission state.

- The source TX_WEN (transmission data write enable) flag is set.
- The DMA transmission request signal (PCMnTDMARQ) is asserted.
- TX_WP_NUM and TX_PHASE are initialized.

1-2) Data is written to the PCMnTXQ register from the DMA controller.

Note The data written to the PCMnTXQ register is then written to the 32-bit x 32-word transmission data FIFO.
The write pointer, TX_WP_NUM, and TX_PHASE are automatically controlled by hardware.

1-3) When the number of words specified by the transmission start timing setting bits (PCMnMOD.Tx_TIM[1:0] bits) have been stored in the transmission data FIFO, the data is transmitted from the PCMnSO pin according to the format of the specified mode.

1-4) When the transmission data FIFO becomes full

1-4-1) When the transmission data FIFO becomes full,

- The source TX_WEN (transmission data write enable) flag is cleared.
- The DMA transmission request signal (PCMnTDMARQ) is deasserted.

1-4-2) When data in the transmission data FIFO has been transmitted from the PCMnSO pin and the transmission data FIFO is no longer full, the state returns to the transmitting state.

- The source TX_WEN (transmission data write enable flag) is enabled.
- The DMA transmission request signal (PCMnTDMARQ) is asserted again.

Writing to the transmission data FIFO is enabled again.

1-4-3) When writing to the transmission data FIFO is performed in the transmission data FIFO full state, an overrun occurs. In this case, the state transitions to the transmission stopped state.

2) If a transmission error occurs, the state transitions to the transmission stopped state.

2-1) If a transmission error[a] occurs, the state transitions to the transmission stopped state.

a) The transmission errors consist of TX_ORE (overrun error), TX_URE (underrun error), and TX_FRE (synchronization error).

Caution When all transmission data has been transmitted from the PCMnSO pin (normal completion), the transmission data FIFO becomes empty, which causes an underrun error. In this case, the state transitions to the transmission stopped state.

The operation in the transmission stopped state is as follows:

- After the last data in the transmission data FIFO has been transmitted before the underrun error occurs, PCMnSO is fixed to 0.
- The transmission error source is retained.
- The values of the transmission data FIFO write counter and read counter are retained.
- Writing to the transmission data FIFO in this state is prohibited.
- The values of TX_WP_NUM and TX_PHASE are retained.

2-2) If a synchronization error (TX_FRE) occurs, clear the source (by setting the PCMnICL.TX_FRECLR bit to 1). If an underrun error (TX_URE) occurs, clear the source (by setting the PCMnICL.TX_URECLR bit to 1).

3) PCMnTEC.TX_ENCLR bit = 1: Transmission disabled

By setting the transmission disable bit to 1, the state returns to the transmission disabled state. All transmission operations stop.

- The source TX_WEN (transmission data write enable flag) is cleared.
- The source TX_ORE (transmission overrun flag) and TX_URE (transmission underrun flag) are cleared.
- The values of the transmission data FIFO write counter and read counter are cleared to 0.

Caution To resume transmission after a transmission underrun or other error occurs, set the state back to the transmission disabled state, and then set it to the transmission enabled state as described below.

- PCMnTEC.TX_ENCLR bit = 1: Transmission disabled
- Wait for one frame period (one word period specified for phase 1 or phase 2, whichever is longer, in mode 5 or 6) after transmission is disabled.
- PCMnTEN.TX_EN bit = 1: Transmission state

This is required because the contents of the transmission data FIFO or the counter value might not be correct due to the error, or the order and contents of the transmission data FIFO, or the order of L-ch data and R-ch data, might have changed in modes 2 to 4.

To change from the transmission enabled state to the transmission disabled state, before transmission resumes, also wait for one frame period (one word period specified for phase 1 or phase 2, whichever is longer, in mode 5 or 6) after transmission is disabled.

Also wait before changing any settings.

However waiting for an additional one-frame period is not required before setting the transmission enabled state after a setting change.

28.5.2 State transitions of reception block

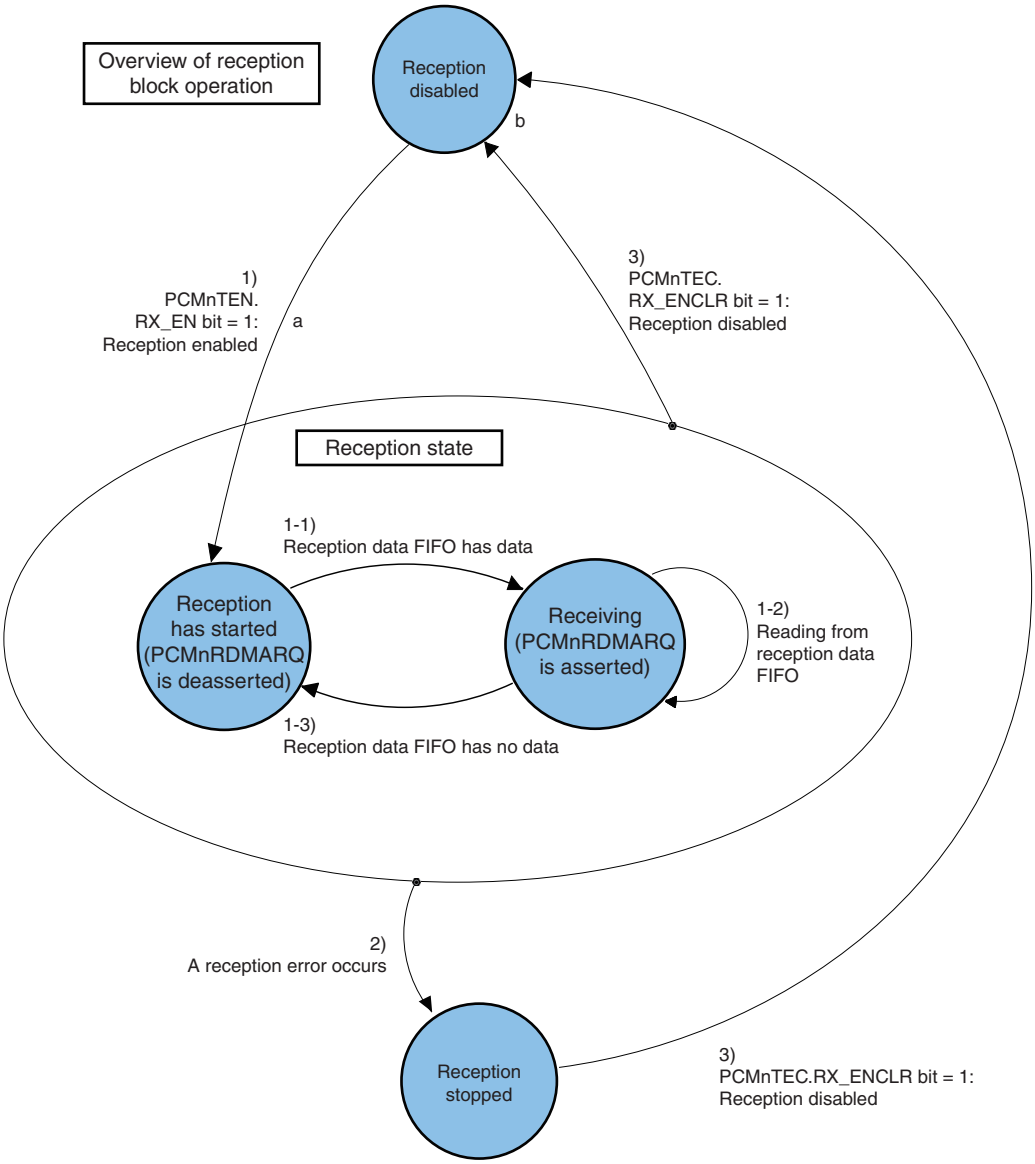


Figure 28-33 State transitions of reception block

a) To enable reception again once reception has been disabled, wait for one frame period (one word period specified for phase 1 or phase 2, whichever is longer, in mode 5 or 6) after reception is disabled.

Waiting is not required, however, after the operation described in b is performed because waiting is performed in the operation described in *b)*.

When operating the microcontroller as a slave, enable reception after the PCMnSEN and PCMnCLK signals from the master are sufficiently stabilized.

b) To change settings after reception stops, wait for one frame period (one word period specified for phase 1 or phase 2, whichever is longer, in mode 5 or 6) after reception is disabled.

Table 28-10 Statuses during reception for each operation source

Operation source	Description	Status		
		Reception disabled	Reception	Reception stopped
PCMnRDMARQ pin	DMA reception request	Fixed to off	Asserted	Fixed to off
PCMnSI pin	Serial data reception	Not receiving	Receiving	Not receiving
RX_REN flag	Reception data write enable flag	Cleared to 0	On or off	On/off status retained
RX_ORE flag	Reception overrun error detected	Cleared to 0	On or off	On/off status retained
RX_URE flag	Reception underrun error detected	Cleared to 0	On or off	On/off status retained
RX_FRE flag	Reception synchronization error detected	Cleared to 0	On or off	On/off status retained
RXFIFO	Reception data FIFO	Read disabled	Read enabled	Read disabled[a]
RX_W_CONT	Write counter of reception data FIFO	0	Counting	Count value retained
RX_R_CONT	Read counter of reception data FIFO	0	Counting	Count value retained
RX_RP_NUM[b]	Word number corresponding to FIFO pointed to by read pointer in reception data FIFO	0	Word number updated	Word number retained
RX_PHASE[b]	Write pointer in reception data FIFO points to either phase 1 or 2	0	Phase display updated	Phase display retained

a) Reading from the reception data FIFO in the reception stopped state might result in a reception underrun error.
b) Fixed to 0 in modes other than modes 5 and 6.

(1) Description of reception block state transitions

The item numbers below generally correspond to those in *Figure 28-33* (although there are some that do not).

1) PCMnTEN.RX_EN bit = 1: Reception state

When operating the microcontroller as a slave, enable reception after the PCMnSEN and PCMnCLK signals from the master are sufficiently stabilized. By setting the reception enable bit to 1, the state transitions to the reception state.

1-1) Serial data is received from the PCMnSI pin.

When one word of data or more is received, it is transmitted to the reception data FIFO.

- The source RX_REN (notification indicating that the reception data FIFO has valid data) flag is set.
- The DMA reception request signal (PCMnRDMARQ) is asserted.
- RX_RP_NUM and RX_PHAS are initialized.

1-2) The DMA controller reads the data stored in the reception data FIFO from the PCMnRXQ register.

The reception data FIFO consists of 32 bits x 32 words.
The read pointer, RX_RP_NUM, and RX_PHASE are automatically controlled by hardware.

1-3) When the reception data FIFO has no more valid data as a result of reading the PCMnRXQ register, the source RX_REN (notification indicating that the reception data FIFO has valid data) flag is cleared,

- The source RX_REN (notification indicating that the reception data FIFO has valid data) flag is cleared.
- The DMA reception request signal (PCMnRDMARQ) is deasserted.
- The state returns to *1-1)* after serial data has been received from the PCMnSI pin and valid data has been stored in the reception data FIFO again.

2) If a transmission error occurs, the state transitions to the reception stopped state.

2-1) If a reception error[a] occurs, the state transitions to the reception stopped state.

a) The reception errors consist of RX_ORE (overrun error), RX_URE (underrun error), and RX_FRE (synchronization error).

The operation in the reception stopped state is as follows:

- Data reception from the PCMnSI pin stops.
- The reception error source is retained.
- The values of the reception data FIFO write counter and read counter are retained.
- Reading from the reception data FIFO in this state is prohibited.
- The values of RX_WP_NUM and RX_PHASE are retained.

2-2) When a synchronization error (RX_FRE) occurs, clear the synchronization error source (by setting the PCMnICL.RX_FRECLR bit to 1).

3) PCMnTEC.RX_ENCLR bit = 1: Reception disabled

By setting the reception disable bit to 1, the state returns to the reception disabled state. All reception operations stop.

- The source RX_REN (reception data FIFO read enable) flag is cleared.
- The source RX_ORE (reception overrun) and RX_URE (reception underrun) flags are cleared.
- The values of the reception data FIFO write counter and read counter are cleared to 0.

Caution To resume reception after a reception error occurs, set the state back to the reception disabled state, and then set it to the reception enabled state as described below.

- PCMnTEC.RX_ENCLR bit = 1: Reception disabled
- Wait for one frame period (one word period specified for phase 1 or phase 2, whichever is longer, in mode 5 or 6) after reception is disabled.
- PCMnTEN.RX_EN bit = 1: Reception state

This is required to initialize the data in the reception data FIFO and the counter value if they are no longer correct as a result of a reception error. In particular, this corrects any change in the order and data in the reception data FIFO or the order of L-ch data and R-ch data in modes 2 to 4.

To change from the reception enabled state to the reception disabled state, before reception resumes, also wait for one frame period (one word period specified for phase 1 or phase 2, whichever is longer, in mode 5 or 6) after reception is disabled.

Also wait before changing any settings.

However waiting for an additional one-frame period is not required before setting the reception enabled state after a setting change.

28.6 Registers

This section describes all PCMn registers.

28.6.1 PCMn register overview

PCMn is controlled and operated by the following registers:

Table 28-11 Control register overview

Register name	Symbol	Address
Operation mode setting register[a]	PCMnMOD	<PCMn_base> + 0000_H
Data transfer enable set register	PCMnTEN	<PCMn_base> + 0004_H
Data transfer enable clear register	PCMnTEC	<PCMn_base> + 0008_H
Data transfer cycle setting register[a]	PCMnCYC1	<PCMn_base> + $000C_H$
Interrupt raw status register	PCMnRAW	<PCMn_base> + 0010_H
Interrupt status register	PCMnSTR	<PCMn_base> + 0014_H
Interrupt enable set register	PCMnIEN	<PCMn_base> + 0018_H
Interrupt enable clear register	PCMnIEC	<PCMn_base> + $001C_H$
Interrupt clear register	PCMnICL	<PCMn_base> + 0020_H
Transmission data register	PCMnTXQ	<PCMn_base> + 0024_H
Reception data register	PCMnRXQ	<PCMn_base> + 0028_H
Data transfer cycle setting register 2[a, b]	PCMnCYC2	<PCMn_base> + 0030_H
PCM signal configuration register	PCMnCFG	<PCMn_base> + 0100_H

a) The settings of the following registers must not be updated when transmission/reception is enabled or stopped, or during the one frame period immediately after transmission/reception is disabled (during the one word period specified for phase 1 or phase 2, whichever is longer, in mode 5 or 6).

- PCMnMOD register
- PCMnCYC1 register
- PCMnCYC2 register

b) The PCMnCYC2 register is valid in modes 5 and 6 only.

<PCMn_base> The <PCMn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

Register access All registers are accessible in word (32-bit) units.
Operation is not guaranteed if these registers are accessed in halfword or byte units. Do not access reserved bits.

28.6.2 PCMn control register details

(1) PCMnMOD - Operation mode setting register

The PCMnMOD register is used to specify the operation mode.

This register is used to select the serial interface timing, master mode or slave mode, transmission start timing, and whether to enable DMA transfer.

Access This register can be read or written in 32-bit units.

Address <PCMn_base> + 0000_H

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24
Reserved							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
Reserved							
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
Reserved							
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
Reserved	Tx_TIM[1:0]		M_S[1:0]		MODE_SEL[2:0]		
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 28-12 PCMnMOD register contents (1/2)

Bit position	Bit name	Function
31 to 7	Reserved	Reserved
6, 5	Tx_TIM[1:0]	Select the number of valid data items in the transmission data FIFO required to trigger serial transfer. **Tx_TIM1 / Tx_TIM0 / Number of valid data items in transmission data FIFO** 0 / 0 / 30 words 0 / 1 / 16 words 1 / 0 / 8 words 1 / 1 / 4 words
4, 3	M_S[1:0]	Select master mode or slave mode. **M_S1 / M_S0 / Selecting master mode or slave mode** 0 / 0 / Stop 0 / 1 / Master mode 1 / 0 / Slave mode 1 / 1 / Setting prohibited

Table 28-12 PCMnMOD register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>2 to 0</td><td>MODE_SEL [2:0]</td><td>Select the serial interface operation mode. For details about the operation modes, see 28.3.2 “Serial interface timing”.
<table>
<tr><th>MODE_ SEL2</th><th>MODE_ SEL1</th><th>MODE_ SEL0</th><th>Selecting operation mode</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Mode 0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Mode 1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Mode 2</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Mode 3</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>Mode 4</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>Mode 5</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>Mode 6</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>Setting prohibited</td></tr>
</table>
</td></tr>
</table>

(2) PCMnTEN - Data transfer enable set register

The PCMnTEN register is used to set the reception enable bit and transmission enable bit.

The reception enable bit and transmission enable bit are used to specify whether to enable serial transfer. When these bits are set to 1, serial transfer is performed. The status can be checked by reading this register.

When 1 is read at the RX_EN bit, reception is enabled.

When 1 is read at the TX_EN bit, transmission is enabled.

When 0 is read, transmission/reception is disabled. Use the PCMnTEC register to clear the reception enable bit and transmission enable bit.

Access This register can be read or written in 32-bit units.

Address <PCMn_base> + 0004_H

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24
Reserved							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
Reserved							
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
Reserved							
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
Reserved						RX_EN	TX_EN
R	R	R	R	R	R	R/W	R/W

Table 28-13 PCMnTEN register contents

Bit position	Bit name	Function
31 to 2	Reserved	Reserved
1	RX_EN	When writing: Specify the reception enable bit. 0: The current state is retained. 1: The reception enable bit is set to 1. When reading: Indicates the value of the reception enable bit. 0: Reception is disabled. 1: Reception is enabled.
0	TX_EN	When writing: Specify the transmission enable bit. 0: The current state is retained. 1: The transmission enable bit is set to 1. When reading: Indicates the value of the transmission enable bit. 0: Transmission disabled. 1: Transmission enabled.

(3) PCMnTEC - Data transfer enable clear register

The PCMnTEC register is used to clear the reception enable bit and transmission enable bit.

The reception enable bit is cleared by writing 1 to the RX_ENCLR bit.

The transmission enable bit is cleared by writing 1 to the TX_ENCLR bit.

When 0 is written, the current state is retained.

Access This register is write-only, in 32-bit units.

Address <PCMn_base> + 0008_H

Initial value Undefined

31	30	29	28	27	26	25	24
Reserved							
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
Reserved							
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
Reserved							
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
Reserved						RX_ENCLR	TX_ENCLR
–	–	–	–	–	–	W	W

Table 28-14 PCMnTEC register contents

Bit position	Bit name	Function
31 to 2	Reserved	Reserved
1	RX_ENCLR	Clear the reception enable bit. 0: The current state is retained. 1: The reception enable bit is cleared to 0.
0	TX_ENCLR	Clear the transmission enable bit. 0: The current state is retained. 1: The transmission enable bit is cleared to 0.

(4) PCMnCYC1 - Data transfer cycle setting register

The PCMnCYC1 register is used to specify transfer-related settings (setting the frame length and the number of valid reception and transmission bits, and choosing whether to enable or disable reception data padding and transmission data padding).

Access This register can be read or written in 32-bit units.

Address <PCMn_base> + $000C_H$

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24
Reserved							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
TX_PD	Reserved		SOB[4:0]				
R/W	R	R	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
RX_PD	Reserved		SIB[4:0]				
R/W	R	R	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
CYC_VAL[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 28-15 PCMnCYC1 register contents (1/2)

Bit position	Bit name	Function
31 to 24	Reserved	Reserved
23	TX_PD	Select whether to enable or disable transmission data padding. 0: Disable 1: Enable To enable padding, the number of valid transmission bits must be set to 8 (SOB[4:0] bit = 07H) or 16 (SOB[4:0] bit = 0FH).
22, 21	Reserved	Reserved
20 to 16	SOB[4:0]	Specify the number of valid transmission bits (PCMnSO signal) as the number of PCMnCLK cycles. The number of valid transmission bits is (specified value + 1). For modes 0 to 4, specify these bits so that (CYC_VAL[7:0] bit value + 1 (the number of valid transmission bits)) are greater than or equal to (SOB[4:0] bit value + 1). Setting range: 07H to 1FH (8 to 32 bits)[a]
15	RX_PD	Specify whether to enable or disable reception data padding. 0: Disable 1: Enable To enable padding, the number of valid reception bits must be set to 8 (SIB[4:0] bit = 07H) or 16 (SIB[4:0] bit = 0FH).

Table 28-15 PCMnCYC1 register contents (2/2)

Bit position	Bit name	Function
14, 13	Reserved	Reserved
12 to 8	SIB[4:0]	Specify the number of valid reception bits (PCMnSI signal) as the number of PCMnCLK cycles. The number of valid reception bits is (specified value + 1). For modes 0 to 4, specify these bits so that (CYC_VAL[7:0] bit value + 1 (the number of valid reception bits)) are greater than or equal to (SIB[4:0] bit value + 1). Setting range: 07H to 1FH (8 to 32 bits)[a]
7 to 0	CYC_VAL [7:0]	For modes 0 to 4, the frame length is specified as the number of PCMnCLK cycles. For mode 5 and 6, the frame length is specified as the number of words specified for phase 1. The frame length is (specified value + 1). For modes 0 to 4, specify these bits so that (CYC_VAL[7:0] bit value + 1) is greater or equal to (SOB[4:0] bit value + 1) (the number of valid transmission bits)) and greater or equal to (SIB[4:0] bit value + 1) (the number of valid reception bits)). The settable number of clock cycles depends on the mode selected. Mode 0 to 4: 07H to 3FH (8 to 64 clock cycles) Modes 5 and 6: 00H to 80H (0 to 128 words)[b]

a) For modes 5 and 6, specify these bits so that the values of the SOB[4:0] and SIB[4:0] bits for simultaneous transmission/reception are the same (SOB[4:0] = SIB[4:0]).
It is possible to specify values in phase 2 (PCMnCYC2.SOB2[4:0] and SIB2[4:0] bits) that differ from the values in phase 1, such that SOB[4:0] is not equal to SOB2[4:0], and SIB[4:0] is not equal to SIB2[4:0].

b) For modes 5 and 6, clearing the PCMnCYC1.CYC_VAL[7:0] bit to 00H is prohibited if the PCMnCYC2.CYC_VAL2[7:0] bits are 00H. If the PCMnCYC1.CYC_VAL[7:0] bits are cleared to 00H, the value of the PCMnCYC2 register becomes valid, and a single-phase operation is performed.
It is possible to specify values in phase 2 (PCMnCYC2.CYC_VAL2[7:0] bits) that differ from the values in phase 1, such that CYC_VAL[7:0] is not equal to CYC_VAL2[7:0].

(5) PCMnRAW - Interrupt raw status register

The PCMnRAW register is used to indicate the status of the reception data FIFO and transmission data FIFO, and transmission and reception errors.

This register indicates the current state, regardless of the setting of the PCMnIEN register.

For details, see *28.3.5 "Data transfer status (status/error/interrupt source)"*.

Access This register is read-only, in 32-bit units.

Address <PCMn_base> + 0010_H

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24
Reserved							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
Reserved							
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
Reserved							
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
RX_RENRAW	RX_ORERAW	RX_URERAW	RX_FRERAW	TX_WENRAW	TX_ORERAW	TX_URERAW	TX_FRERAW
R	R	R	R	R	R	R	R

Table 28-16 PCMnRAW register contents (1/2)

Bit position	Bit name	Function
31 to 8	Reserved	Reserved
7	RX_RENRAW	Indicates the raw status of the reception data FIFO. 0: There is no valid reception data (reading disabled). 1: There is valid reception data (reading enabled).
6	RX_ORERAW	Indicates the raw status of the reception overrun error. 0: Normal 1: An overrun error was detected.
5	RX_URERAW	Indicates the raw status of the reception underrun error. 0: Normal 1: Reception underrun error detected
4	RX_FRERAW	Indicates the raw status of the reception frame synchronization error. 0: Normal 1: A frame synchronization error was detected (only during slave operation).
3	TX_WENRAW	Indicates the raw status of the transmission data FIFO. 0: The transmission data FIFO has no free space (writing disabled). 1: The transmission data FIFO has free space (writing enabled).

Table 28-16 PCMnRAW register contents (2/2)

Bit position	Bit name	Function
2	TX_ORERAW	Indicates the raw status of the transmission overrun error. 0: Normal 1: An overrun error was detected.
1	TX_URERAW	Indicates the raw status of the transmission underrun error. 0: Normal 1: Reception underrun error detected
0	TX_FRERAW	Indicates the raw status of the transmission frame synchronization error. 0: Normal 1: A frame synchronization error was detected (only during slave operation).

(6) PCMnSTR - Interrupt status register

The PCMnSTR register is used to check the interrupt status.

The value of this register is the result of masking the PCMnRAW register using the value of the PCMnIEN register.

Access This register is read-only, in 32-bit units.

Address <PCMn_base> + 0014_H

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24
Reserved							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
Reserved							
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
Reserved							
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
RX_REN	RX_ORE	RX_URE	RX_FRE	TX_WEN	TX_ORE	TX_URE	TX_FRE
R	R	R	R	R	R	R	R

Table 28-17 PCMnSTR register contents (1/2)

Bit position	Bit name	Function
31 to 8	Reserved	Reserved
7	RX_REN	Indicates the reception data FIFO interrupt status. 0: No interrupt request was issued. 1: An interrupt request was issued.
6	RX_ORE	Indicates the reception overrun error interrupt status. 0: No interrupt request was issued. 1: An interrupt request was issued.
5	RX_URE	Indicates the reception underrun error interrupt status. 0: No interrupt request was issued. 1: An interrupt request was issued.
4	RX_FRE	Indicates the reception frame synchronization error interrupt status. 0: No interrupt request was issued. 1: An interrupt request was issued.
3	TX_WEN	Indicates the transmission data FIFO interrupt status. 0: No interrupt request was issued. 1: An interrupt request was issued.

Table 28-17 PCMnSTR register contents (2/2)

Bit position	Bit name	Function
2	TX_ORE	Indicates the transmission overrun error interrupt status. 0: No interrupt request was issued. 1: An interrupt request was issued.
1	TX_URE	Indicates the transmission underrun error interrupt status. 0: No interrupt request was issued. 1: An interrupt request was issued.
0	TX_FRE	Indicates the transmission frame synchronization error interrupt status. 0: No interrupt request was issued. 1: An interrupt request was issued.

(7) PCMnIEN - Interrupt enable set register

The PCMnIEN register is used to enable interrupt request issuance for each interrupt source.

The setting of the interrupt enable bits can be checked by reading this register.

Access This register can be read or written in 32-bit units.

Address <PCMn_base> + 0018_H

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24
Reserved							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
Reserved							
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
Reserved							
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
RX_REN_EN	RX_ORE_EN	RX_URE_EN	RX_FRE_EN	TX_WEN_EN	TX_ORE_EN	TX_URE_EN	TX_FRE_EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 28-18 PCMnIEN register contents (1/2)

Bit position	Bit name	Function
31 to 8	Reserved	Reserved
7	RX_REN_EN	Specify whether to enable the reception overrun error interrupt. When writing: Set the interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is set. When reading: The current setting of the interrupt enable bit is retained. 0: Generating interrupt requests is disabled. 1: Generating interrupt requests is enabled.
6	RX_ORE_EN	Specify whether to enable the reception overrun error interrupt. When writing: Set the interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is set. When reading: The current setting of the interrupt enable bit is retained. 0: Generating interrupt requests is disabled. 1: Generating interrupt requests is enabled.
5	RX_URE_EN	Specify whether to enable the reception underrun error interrupt. When writing: Set the interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is set. When reading: The current setting of the interrupt enable bit is retained. 0: Generating interrupt requests is disabled. 1: Generating interrupt requests is enabled.

Table 28-18 PCMnIEN register contents (2/2)

Bit position	Bit name	Function
4	RX_FRE_EN	Specify whether to enable the reception frame synchronization error interrupt. When writing: Set the interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is set. When reading: The current setting of the interrupt enable bit is retained. 0: Generating interrupt requests is disabled. 1: Generating interrupt requests is enabled.
3	TX_WEN_EN	Specify whether to enable the transmission data FIFO interrupt. When writing: Set the interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is set. When reading: The current setting of the interrupt enable bit is retained. 0: Generating interrupt requests is disabled. 1: Generating interrupt requests is enabled.
2	TX_ORE_EN	Specify whether to enable the transmission overrun error interrupt. When writing: Set the interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is set. When reading: The current setting of the interrupt enable bit is retained. 0: Generating interrupt requests is disabled. 1: Generating interrupt requests is enabled.
1	TX_URE_EN	Specify whether to enable the transmission underrun error interrupt. When writing: Set the interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is set. When reading: The current setting of the interrupt enable bit is retained. 0: Generating interrupt requests is disabled. 1: Generating interrupt requests is enabled.
0	TX_FRE_EN	Specify whether to enable the transmission frame synchronization error interrupt. When writing: When writing: Set the interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is set. When reading: The current setting of the interrupt enable bit is retained. 0: Generating interrupt requests is disabled. 1: Generating interrupt requests is enabled.

(8) PCMnIEC - Interrupt enable clear register

The PCMnIEC register is used to clear the interrupt request issuance enable bit (interrupt enable bit) for each interrupt source.

Access This register is write-only, in 32-bit units.

Address <PCMn_base> + $001C_H$

Initial value Undefined

31	30	29	28	27	26	25	24
Reserved							
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
Reserved							
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
Reserved							
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
RX_REN_MASK	RX_ORE_MASK	RX_URE_MASK	RX_FRE_MASK	TX_WEN_MASK	TX_ORE_MASK	TX_URE_MASK	TX_FRE_MASK
W	W	W	W	W	W	W	W

Table 28-19 PCMnIEC register contents (1/2)

Bit position	Bit name	Function
31 to 8	Reserved	Reserved
7	RX_REN_MASK	Clear the reception data FIFO interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is cleared (the interrupt is masked).
6	RX_URE_MASK	Clear the reception overrun error interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is cleared (the interrupt is masked).
5	RX_URE_MASK	Clear the reception underrun error interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is cleared (the interrupt is masked).
4	RX_FRE_MASK	Clear the reception frame synchronization error interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is cleared (the interrupt is masked).
3	TX_WEN_MASK	Clear the transmission data FIFO interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is cleared (the interrupt is masked).

Table 28-19 PCMnIEC register contents (2/2)

Bit position	Bit name	Function
2	TX_ORE_ MASK	Clear the transmission overrun error interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is cleared (the interrupt is masked).
1	TX_URE_ MASK	Clear the transmission underrun error interrupt enable bit. 0:The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is cleared (the interrupt is masked).
0	TX_FRE_ MASK	Clear the transmission frame synchronization error interrupt enable bit. 0: The current setting of the interrupt enable bit is retained. 1: The interrupt enable bit is cleared (the interrupt is masked).

(9) PCMnICL - Interrupt clear register

The PCMnICL register is used to clear specific interrupt sources.

Other interrupt sources can be cleared by disabling transmission (PCMnTEC.TX_ENCLR = 1) or reception (PCMnTEC.RX_ENCLR = 1).

Access This register is write-only, in 32-bit units.

Address <PCMn_base> + 0020_H

Initial value Undefined

31	30	29	28	27	26	25	24
Reserved							
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
Reserved							
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
Reserved							
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
Reserved			RX_FRECLR	Reserved		TX_URECLR	TX_FRECLR
–	–	–	W	–	–	W	W

Table 28-20 PCMnICL register contents

Bit position	Bit name	Function
31 to 5	Reserved	Reserved
4	RX_FRECLR	Clear the RX_FRE source (reception synchronization error) bit. 0: The interrupt source is retained. 1: The interrupt source is cleared.
3, 2	Reserved	Reserved
1	TX_URECLR	Clear the TX_URE source (transmission underrun error) bit. 0: The interrupt source is retained. 1: The interrupt source is cleared.
0	TX_FRECLR	Clear the TX_FRE source (transmission synchronization error) bit. 0: The interrupt source is retained. 1: The interrupt source is cleared.

(10) PCMnTXQ - Transmission data register

The PCMnTXQ register is used to specify data to be writtten to the transmission data FIFO.

The last data written to the transmission data FIFO can be read by reading this register.

Access This register can be read or written in 32-bit units.

Address <PCMn_base> + 0024_H

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24
TXQ[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
TXQ[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
TXQ[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
TXQ[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 28-21 PCMnTXQ register contents

Bit position	Bit name	Function
31 to 0	TXQ[31:0]	Specify data to be writtten to the transmission data FIFO.

When data is written to the PCMnTXQ register, the data is written to the entry pointed to by the write pointer in the 32-bit x 32-word transmission data FIFO. The write pointer is automatically controlled by hardware.

When the PCMnTXQ register is read, the last written data is read.

An example of writing data to the transmission data FIFO is provided below.

Data longer than the transmission data bit length is ignored (x: Don’t Care).

Example Writing xxxx_x067H by using the PCMnTXQ register

The figure below shows an example of writing data to the transmission data FIFO as follows:

Transmission data bit length = 10 bits (PCMnCYC1.SOB[4:0] bits = 09H)

Writing xxxx_x123H by using the PCMnTXQ register

Writing xxxx_x245H by using the PCMnTXQ register

Figure 28-34 Example of writing data to transmission data FIFO

Caution The figure above shows an example of writing with no data padding.
When writing using data padding (even if the transmission data bit length is 8 or 16 bits), 32-bit data is padded. (No Don't-Care bits are used.)
For details, see *28.3.3 "Data padding"*.

(11) PCMnRXQ - Reception data register

The PCMnRXQ register is used to read received data from the reception data FIFO. When reception is not performed, the value of this register is 0000_0000H.

Access This register is read-only, in 32-bit units.

Address <PCMn_base> + 0028_H

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24
RXQ[31:24]							
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
RXQ[23:16]							
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
RXQ[15:8]							
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
RXQ[7:0]							
R	R	R	R	R	R	R	R

Table 28-22 PCMnRXQ register contents

Bit position	Bit name	Function
31 to 0	RXQ[31:0]	Indicates data read from the reception data FIFO.

Data is read from the entry pointed to by the read pointer in the 32-bit x 32-word reception data FIFO. The read pointer is automatically controlled by hardware.

No data can be written to the PCMnRXQ register.

An example of reading data from the reception data FIFO is provided below.

Data beyond the reception data bit length is cleared to 0.

Example The figure below shows an example of writing data to the transmission data FIFO as follows:

Reception data bit length = 28 bits (PCMnCYC1.SIB[4:0] bits = 1BH)

Reading data (0123_4567H) by using the PCMnRXQ register

Reading data (089A_BCDEH) by using the PCMnRXQ register

Reading data (0F24_68ACH) by using the PCMnRXQ register

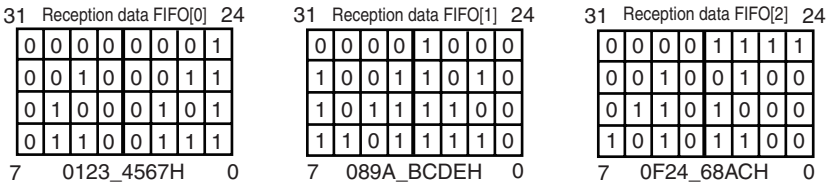


Figure 28-35 Example of reading data from reception data FIFO

Caution The figure above shows an example of reading with no data padding.
When reading using data padding (even if the reception data bit length is 8 or 16 bits), 32-bit data is padded.
For details, see *28.3.3 "Data padding"*.

(12) PCMnCYC2 - Data transfer cycle setting register2

The PCMnCYC2 register is used to specify settings for phase 2 in mode 5 or 6.

The settings of this register are invalid in other modes.

The settings of the PCMnCYC2 register must not be updated when transmission/reception is enabled or stopped, or during the one word period specified for phase 1 or phase 2, whichever is longer, after transmission/reception is disabled.

Access This register can be read or written in 32-bit units.

Address <PCMn_base> + 0030_H

Initial value $0000\ 0000_H$

31	30	29	28	27	26	25	24
Reserved							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
Reserved			SOB2[4:0]				
R	R	R	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
Reserved			SIB2[4:0]				
R	R	R	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
CYC_VAL2[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 28-23 PCMnCYC2 register contents

Bit position	Bit name	Function
31 to 21	Reserved	Reserved
20 to 16	SOB2[4:0]	Specify the number of valid transmission bits (PCMnSO signal) as the number of PCMnCLK cycles. The number of valid transmission bits is (specified value + 1). Setting range: 07H to 1FH (8 to 32 bits)[a]
15 to 13	Reserved	Reserved
12 to 8	SIB2[4:0]	Specify the number of valid reception bits (PCMnSI signal) as the number of PCMnCLK cycles. The number of valid reception bits is (specified value + 1). Setting range: 07H to 1FH (8 to 32 bits)[a]
7 to 0	CYC_VAL2 [7:0]	Specify the number of words per frame for phase 2. In mode 5: 00H to 80H (0 to 128 words)[b] In mode 6: 00H to 80H (0 to 128 words)[b]

a) For modes 5 and 6, specify these bits so that the values of the SOB2[4:0] and SIB2[4:0] bits for simultaneous transmission/reception are the same (SOB2[4:0] = SIB2[4:0]).
A value (such as SOB[4:0] not equal SOB2[4:0] or SIB[4:0] not equal SIB2[4:0]) different from phase 1 (PCMnCYC1.SOB[4:0] and SIB[4:0] bits) can be specified.

b) For modes 5 and 6, clearing the PCMnCYC2.CYC_VAL2[7:0] bits to 00H is prohibited if the PCMnCYC1.CYC_VAL[7:0] bits are 00H.
If the CYC_VAL2[7:0] bits are cleared to 00H, the value of the PCMnCYC1 register becomes valid, and a single-phase operation is performed.
A value (such as CYC_VAL[7:0] bit value not equal CYC_VAL2[7:0] bit value) different from phase 1 (CYC_VAL[7:0] bits) can be specified.

(13) PCMnCFG - PCM signal configuration register

The PCMnCFG register is used to specify the timing at which the PCM signals are output and sampled, and the active level of the PCMnCLK signal.

<R> **Access** This register can be read or written in 32-bit units.

Address <PCMn_base> + 0100_H

<R> **Initial value** $0000\ 0003_H$. This register is initialized by any reset.

<R>

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

<R>

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	0	0	0	0	0	PCMn CFGC	PCMn CFGS
R	R	R	R	R	R	R/W	R/W

<R> **Table 28-24 PCMnCFG register contents**

Bit position	Bit name	Function
1	PCMnCFGC	Specify the timing at which the PCM signals are output and sampled (see *Table 28-25*).

The timing at which the PCMnSEN, PCMnSO, and PCMnSI signals are output and sampled, which is specified by setting PCMnCLK, is listed in the following table:

Table 28-25 Timing at which the PCM signals are output or sampled

Mode	PCMnCFGC	PCMnSEN output timing (in master mode)	PCMnSO output timing	PCMnSEN (in slave mode) and PCMnSI input sampling timing
Mode 0	1	PCMnCLK rising edge		PCMnCLK falling edge
	0	PCMnCLK falling edge		PCMnCLK rising edge
Mode 1	1	PCMnCLK falling edge	PCMnCLK rising edge	
	0	PCMnCLK rising edge	PCMnCLK falling edge	
Mode 2	1	PCMnCLK falling edge		PCMnCLK rising edge
	0	PCMnCLK rising edge		PCMnCLK falling edge
Mode 3	1	PCMnCLK falling edge		PCMnCLK rising edge
	0	PCMnCLK rising edge		PCMnCLK falling edge
Mode 4	1	PCMnCLK falling edge		PCMnCLK rising edge
	0	PCMnCLK rising edge		PCMnCLK falling edge

Table 28-25 Timing at which the PCM signals are output or sampled

Mode	PCMnCFGC	PCMnSEN output timing (in master mode)	PCMnSO output timing	PCMnSEN (in slave mode) and PCMnSI input sampling timing
Mode 5	1	PCMnCLK rising edge		PCMnCLK falling edge
	0	PCMnCLK falling edge		PCMnCLK rising edge
Mode 6	1	PCMnCLK falling edge	PCMnCLK rising edge	
	0	PCMnCLK rising edge	PCMnCLK falling edge	

Chapter 29 Media Local Bus (MLB)

This chapter contains a block diagram and a list of all media local bus (MLB) registers. For detailed information about the functionality and the registers, see the following document:

OS62400 MediaLB Device Interface Macro

Document number: DS62400AP5

SMSC®

The first section describes all properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names.

29.1 V850E2/Sx4-H MLB Features

Instances

This microcontroller has one instance of the media local bus, an interface connected to the MOST controller.

Table 29-1 Instances of MLB

MLB	
Number of instances	1
Name	MLB0

Instances index n

Throughout this chapter, the individual instances of MLB are identified by the index “n” (n = 0), for example, MLBnDCCR for the MLBn device control configuration register.

Channel index m

Throughout this chapter, the individual channels are indentified by the index “m” (m = 0 to 30), for example, MLBnCECRm for the channel m entry configuration register.

Register addresses

All MLBn register addresses are given as addresses offset from the individual base address <MLBn_base>. The base address <MLBn_base> of each MLBn is listed in the following table:

Table 29-2 Register base addresses <MLBn_base>

MLBn	<MLBAn_base> address
MLB0	FF82 $D000_H$

Clock supply

The following clock is supplied to the MLB module:

Table 29-3 MLBn clock supply

MLBn	Clock	Connected to:
MLB0	ram_clk/sys_clk[a]	Clock generator CKSCLK_124

a) A clock divided by 2 is input for sys_clk. If fs is1,024, input a clock with a frequency of 100 MHz or more for ram_clk.

Interrupts and DMA

The MLB can generate the following interrupt requests and DMA requests:

Table 29-4 MLBn interrupt requests and DMA requests

MLBn signals	Function	Connected to:
mlb_cint	Channel interrupt	• Interrupt controller INTMLB0CI • DMA controller trigger 99
mlb_sint	System interrupt	• Interrupt controller INTMLB0SI • DMA controller trigger 100
mlb_dint[30:0]	Data interrupt	Not connected

MLB hardware reset

The MLB and its registers are initialized by the following reset signal:

Table 29-5 MLBn reset signal

MLBn	Reset signal
MLB0	System reset SYSRES

I/O signals

The I/O signals of the MLB are listed in the following table:

Table 29-6 MLBn I/O signals

MLBn signals	Function	Connected to:
MLB0:		
mlbclk_in	MediaLB clock input	Port MLBA0CLK
mlbsig_in	MediaLB signal information I/O	Port MLBA0SIG
mlbsig_out		
mlbsig_oe		
mlbdat_in	MediaLB data I/O	Port MLBA0DAT
mlbdat_out		
mlbdat_oe		
mlb_5oin		Not connected

29.2 Functional Overview

The block diagram of the MLB is shown below.

For details about the functionality and the registers, see the following document:

OS62400 MediaLB Device Interface Macro

Document number: DS62400AP5

SMSC

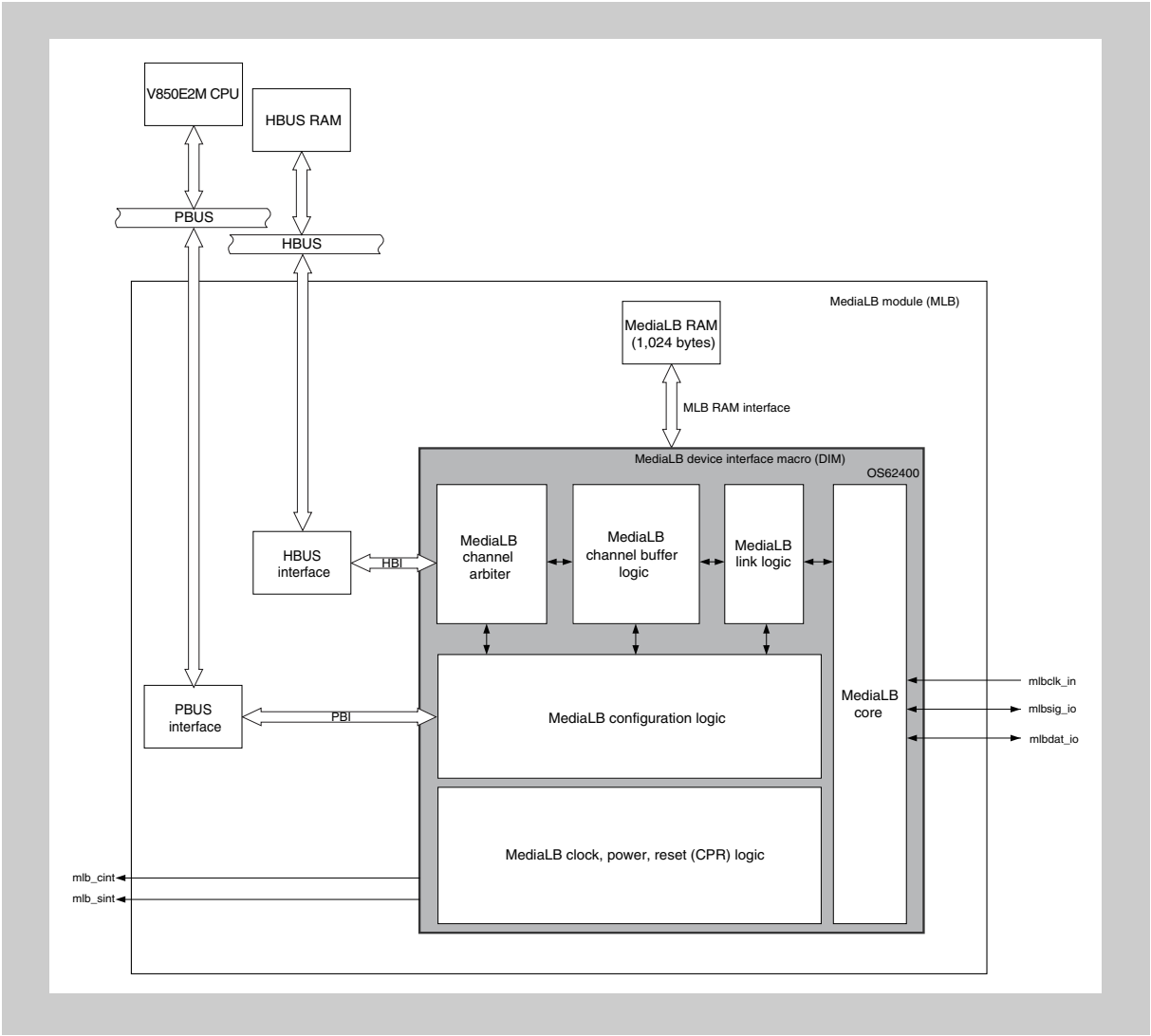


Figure 29-1 Block diagram of the media local bus interface

29.3 MLBn Register Overview

This section contains a list of the media local bus registers.

MLB is controlled and operated by the following registers:

Table 29-7 MBLn register overview

Register name	Symbol	Address
Configuration control registers		
Device control configuration register	MLBnDCCR	<MLBn_base> + 0000_H
System status configuration register	MLBnSSCR	<MLBn_base> + 0004_H
System data configuration register	MLBnSDCR	<MLBn_base> + 0008_H
System mask configuration register	MLBnSMCR	<MLBn_base> + $000C_H$
Version control configuration register	MLBnVCCR	<MLBn_base> + $001C_H$
Synchronous base address configuration register	MLBnSBCR	<MLBn_base> + 0020_H
Asynchronous base address configuration register	MLBnABCR	<MLBn_base> + 0024_H
Control base address configuration register	MLBnCBCR	<MLBn_base> + 0028_H
Isochronous base address configuration register	MLBnIBCR	<MLBn_base> + $002C_H$
Channel interrupt configuration register	MLBnCICR	<MLBn_base> + 0030_H
Channel m configuration registers		
Channel m entry configuration register	MLBnCECRm	<MLBn_base> + 0040_H + m x 10
Channel m status configuration register	MLBnCSCRm	<MLBn_base> + 0044_H + m x 10
Channel m current buffer configuration register	MLBnCCBCRm	<MLBn_base> + 0048_H + m x 10
Channel m next buffer configuration register	MLBnCNBCRm	<MLBn_base> + $004C_H$ + m x 10
Local channel m buffer configuration register	MLBnLCBCRm	<MLBn_base> + 0280_H + m x 4

<MLBn_base> The base addresses <MLBn_base> of the MLBn is defined in the first section of this chapter under the key word “Register addresses”.

Chapter 30 IEBus Controller (IEBB)

IEBus (Inter Equipment Bus) is a small-scale digital data transfer system that transfers data between units. To implement IEBus with the V850E2/Sx4-H, an external IEBus driver and receiver are necessary because they are not provided.

The internal IEBus controllers of the V850E2/Sx4-H are of negative logic.

30.1 V850E2/Sx4-H IEBB Features

Instances

This microcontroller has the following number of instances of the IEBB.

Table 30-1 Instances of IEBB

IEBB	
Number of instances	1
Name	IEBB0

Instances index n

Throughout this chapter, the individual instances of IEBB are identified by the index “n” (n = 0), for example, IEBBnBCR for the IEBBn bus control register.

Register addresses

All IEBBn register addresses are given as addresses offset from the individual base address <IEBBn_base>.

The base address <IEBBn_base> of each IEBBn is listed in the folowing table:

Table 30-2 Register base address <IEBBn_base>

IEBBn	<IEBBn_base>
IEBB0	FF82 $C000_H$

Clock supply

The following clock is supplied to IEBBn:

Table 30-3 IEBBn clock supply

IEBBn	Clock	Connected to:
IEBB0	PCLK	Clock generator CKSCLK_106

Interrupts and DMA IEBB can generate the following interrupt requests and DMA requests:

Table 30-4 IEBBn interrupts and DMA requests

Interrupt request signal	Function	Connected to:
IEBBTD	Data interrupt request	Interrupt controller INTIEBB0D DMA controller trigger 85
IEBBTV	Vector interrupt request	Interrupt controller INTIEBB0V DMA controller trigger 86
IEBBTERR	Error interrupt request	Interrupt controller INTIEBB0ERR
IEBBTSTA	Status interrupt request	Interrupt controller INTIEBB0STA

IEBB hardware reset IEBB0 and its register are initialized by the following reset signal:

Table 30-5 IEBBn reset signal

IEBBn	Reset signal
IEBBn	System reset SYSRES

I/O signals The I/O signals of IEBB0 are listed in the following table:

Table 30-6 IEBBn I/O signals

IEBBn signal	Function	Connected to:
IEBB0:		
IEBBTRXB	IEBB0 reception data	Port $\overline{\text{IEBB0RX}}$
IEBBTTXB	IEBB0 transmission data	Port $\overline{\text{IEBB0TX}}$

30.2 Configuration

30.2.1 Function overview

Features summary

- **The data transfer system complies with the IEBus (communication mode 1/communication mode 2) protocol.**
- **The IEBus (Inter Equipment Bus) controller is mainly intended to transfer data between automotive devices by using a two-line serial bus interface.**
- Effective transmission speed: Approximately 17 kbps (mode 1), or approximately 26 kbps (mode 2)
- The single mode or FIFO mode can be selected.
- Maximum number of transferred bytes: 32 bytes/frame (mode 1)
  128 bytes/frame (mode 2)
- To implement IEBus, an external IEBus driver and receiver are necessary. The driver and receiver are not built in.
- The internal IEBus controller uses negative logic.
- Operation clock: 8 MHz
- Interrupt request signal
  - Data interrupt (IEBBTD)
    For transmission data write processing (single mode, FIFO mode)
    For reception data read processing (single mode)
  - Error interrupt (IEBBTERR)
    For error processing
  - Status interrupt (IEBBTSTA)
    Start interrupt processing (single mode)
    Status transmission interrupt (single mode)
    Communication completion interrupt (single mode, FIFO mode)
    Frame completion interrupt (single mode, FIFO mode)
  - Vector interrupt (IEBBTV)
    Occurs at the same time as IEBBTERR or IEBBTSTA (single mode)
    For reception data read processing (FIFO mode)
- Pin configuration
  - IEBBTRXB: IEBus reception data input signal
  - IEBBTTXB: IEBus transmission data output signal

30.2.2 Block diagram

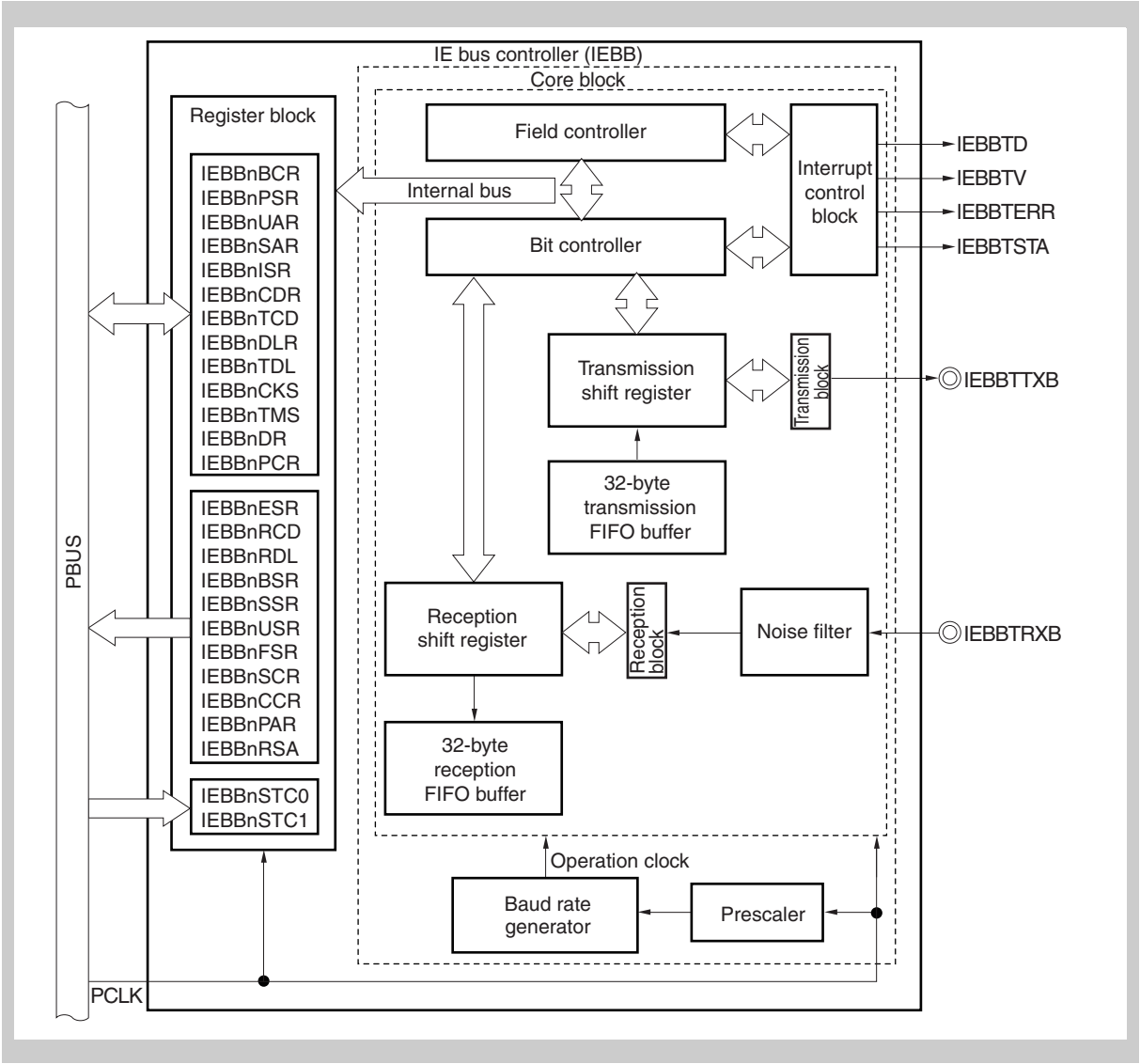


Figure 30-1 IEBBn block diagram

30.2.3 Functional differences between V850E2/Sx4-H and V850ES/Sx3

The functional differences between V850E2/Sx4-H and V850ES/Sx3 are shown below.

Table 30-7 Functional differences between V850E2/Sx4-H and V850ES/Sx3

Function	V850E2/Sx4-H	V850ES/Sx3	Remark
Supply clock	8 MHz	6.29 MHz (6 MHz)	
FIFOs (32-byte transmission FIFO buffer, 32-byte reception FIFO buffer)	Available	Not available	
Function for automatically reissuing master requests when arbitration loss occurs	Available	Not available	See Caution 1.
Function for automatically responding to slave status requests	Available	Not available	See Caution 2.

Cautions

1. Note that, for the function for automatically reissuing master requests when arbitration loss occurs, there are differences between V850E2/Sx4-H and V850ES/Sx3.

- V850E2/Sx4-H
  - In the single mode
    When arbitration loss occurs, a master request must be reissued by using software.
  - In the FIFO mode
    When arbitration loss occurs, a master request must be reissued by using hardware. However, if arbitration losses consecutively occur the number of times specified by the IEBBnTMS.IEBBnALC2 to IEBBnALC0 bits, the request must be reissued by using software.
- V850ES/Sx3

  When arbitration loss occurs, a master request must be reissued by using software.

2. Note that, for the function for automatically responding to slave status requests, there are differences between V850E2/Sx4-H and V850ES/Sx3.

- V850E2/Sx4-H
  - In the single mode
    If there is a slave status request (control data: 0H, 6H), during the processing for the generated status interrupt (IEBBTSTA, IEBBTV), read the IEBBn slave status register (IEBBnSSR) and write to the IEBBn data register (IEBBnDR).
  - In the FIFO mode
    Use the hardware to write the value of the IEBBnSSR register to the IEBBnDR register.
- V850ES/Sx3
  - If there is a slave status request (control data: 0H, 6H), during the processing for the generated status interrupt (INTSTA, INTIE2), read the IEBBn slave status register (SSR) and write to the IEBBn data register (DR).

30.3 Registers

30.3.1 IEBBn register overview

IEBBn is controlled and operated by thefollowing registers:

Table 30-8 IEBBn registers

Register name	Symbol	Address
IEBBn bus control register	IEBBnBCR	<IEBBn_base> + 0000_H
IEBBn power save register	IEBBnPSR	<IEBBn_base> + 0004_H
IEBBn unit address register	IEBBnUAR	<IEBBn_base> + 0008_H
IEBBn slave address register	IEBBnSAR	<IEBBn_base> + $000C_H$
IEBBn partner address register	IEBBnPAR	<IEBBn_base> + 0010_H
IEBBn reception slave address register	IEBBnRSA	<IEBBn_base> + 0014_H
IEBBn control data register	IEBBnCDR	<IEBBn_base> + 0018_H
IEBBn transmission control data register	IEBBnTCD	<IEBBn_base> + $001C_H$
IEBBn reception control data register	IEBBnRCD	<IEBBn_base> + 0020_H
IEBBn message length register	IEBBnDLR	<IEBBn_base> + 0024_H
IEBBn transmission message length register	IEBBnTDL	<IEBBn_base> + 0028_H
IEBBn reception message length register	IEBBnRDL	<IEBBn_base> + $002C_H$
IEBBn clock selection register	IEBBnCKS	<IEBBn_base> + 0030_H
IEBBn transfer mode setting register	IEBBnTMS	<IEBBn_base> + 0034_H
IEBBn pointer clear register	IEBBnPCR	<IEBBn_base> + 0038_H
IEBBn buffer status register	IEBBnBSR	<IEBBn_base> + $003C_H$
IEBBn slave status register	IEBBnSSR	<IEBBn_base> + 0040_H
IEBBn unit status register	IEBBnUSR	<IEBBn_base> + 0044_H
IEBBn interrupt status register	IEBBnISR	<IEBBn_base> + 0048_H
IEBBn error status register	IEBBnESR	<IEBBn_base> + $004C_H$
IEBBn field status register	IEBBnFSR	<IEBBn_base> + 0050_H
IEBBn success count register	IEBBnSCR	<IEBBn_base> + 0054_H
IEBBn communication count register	IEBBnCCR	<IEBBn_base> + 0058_H
IEBBn status clear register 0	IEBBnSTC0	<IEBBn_base> + $005C_H$
IEBBn status clear register 1	IEBBnSTC1	<IEBBn_base> + 0060_H
IEBBn data register	IEBBnDR	<IEBBn_base> + 0064_H

<IEBBn_base> The IEBBn base address <IEBBn_base> is defined in *Table 30-2 "Register base address <IEBBn_base>"*.

30.3.2 IEBBn control register details

(1) IEBBnBCR - IEBBn bus control register

The IEBBnBCR register is used to control the operation of IEBBn.

Access This register can be read or written in 8-bit or 1-bit units.

Address <IEBBn_base> + 0000H

Initial value 00H

The IEBBnMSRQ, IEBBnALRQ, IEBBnSTXE, and IEBBnSRXE bits are reset by writing 0 to the IEBBnPW bit.

Cautions

1. When operation is enabled (when the IEBBnPW bit = 1), writing 1 to the IEBBnMSRQ bit is prohibited while the IEBBnMSRQ bit = 1. To write 1 to this bit, first clear it to 0.
2. Note the following when accessing the register:
   - When the IEBBnPW bit = 0, it is not possible to write to the IEBBnMSRQ, IEBBnALRQ, IEBBnSTXE, and IEBBnSRXE bits.
   - Because the IEBBnMSRQ, IEBBnALRQ, IEBBnSTXE, and IEBBnSRXE bits are reset at the same time by writing 0 to the IEBBnPW bit, even if an 8-bit write that results in the IEBBnPW bit being cleared to 0 is performed, the IEBBnMSRQ, IEBBnALRQ, IEBBnSTXE, and IEBBnSRXE bits are not written to. When an 8-bit write that results in the IEBBnPW bit being set to 1 is performed, the IEBBnMSRQ, IEBBnALRQ, IEBBnSTXE, and IEBBnSRXE bits can be written to.

     Example: If 78H is written to the IEBBnBCR register and then the register is read, 00H is returned.

     If F8H is written to the IEBBnBCR register and then the register is read, F8H is returned.
   - When a one-bit manipulation instruction is used to write to the IEBBnMSRQ, IEBBnALRQ, IEBBnSTXE, and IEBBnSRXE bits while the IEBBnMSRQ bit = 1, there might be contention between the clearing of the IEBBnMSRQ bit by the hardware and the setting of the bit by the hardware. Therefore, such writing is prohibited.

7	6	5	4	3	2	1	0
IEBBn PW	IEBBn MSRQ	IEBBn ALRQ	IEBBn STXE	IEBBn SRXE	0	0	0
R/W	R/W	R/W	R/W	R/W	R	R	R

Table 30-9 IEBBnBCR register contents

Bit position	Bit name	Function
7	IEBBnPW	Communication enable flag 0: Stop IEBBn unit operation. 1: Enable IEBBn unit operation. **Caution** When the IEBBnPW bit is set (to 1), the IEBBnPSR, IEBBnUAR, IEBBnCKS, and IEBBnTMS registers below cannot be overwritten. Therefore, these registers must be set up before setting the IEBBnPW bit.
6	IEBBnMSRQ	Master request flag 0: Do not request the IEBBn unit as the master. 1: Request the IEBBn unit as the master.
5	IEBBnALRQ	Broadcast request flag 0: Request individual communication. 1: Request broadcast communication.
4	IEBBnSTXE	Slave transmission enable flag 0: Disable slave transmission. 1: Enable slave transmission.
3	IEBBnSRXE	Slave reception enable flag 0: Disable slave reception. 1: Enable slave reception.

(a) Communication enable flag (IEBBnPW): Bit 7

- Set/clear condition

  Set: By software (Write 1 to the IEBBnPW bit.)
  Clear: By software (Write 0 to the IEBBnPW bit.)

Depending on when the IEBBnPW bit is set (to 1), the IEBBn communication participation method differs.

Table 30-10 IEBBnPW bit setting timing and communication participation method

Timing for setting the IEBBnPW bit (to 1)	IEBBn communication participation method
When communication is not being performed on IEBus	Communication is participated in starting at the next frame or communication is started.
When communication is being performed on IEBus, and start bit communication is being performed by another bus master	Participates in communication from that frame if the start bit is detected. If the start bit is not detected, participates in communication from the next frame.
When communication is being performed on IEBus, and post-start bit communication is being performed by another bus master	Participates in communication from the next frame.

If the IEBBnPW bit is cleared (to 0), communication is immediately stopped even if it is in progress, and the internal flags and registers are reset, with some exceptions. The registers that are not reset by the IEBBnPW bit are shown below.

When the IEBBnPW = 0, even if another unit starts communication, IEBBn does not respond.

Table 30-11 Registers that are not reset by the IEBBnPW bit

Registers that are not reset by the IEBBnPW bit	Remark
EBBnPSR	Not reset
IEBBnUAR	Not reset
IEBBnSAR	Not reset
IEBBnCDR	Data written from the CPU is not reset but data received during communication is.
IEBBnTCD	Not reset
IEBBnDLR	Data written from the CPU is not reset but data received during communication is.
IEBBnTDL	Not reset
IEBBnCKS	Not reset
IEBBnTMS	Not reset
IEBBnPCR	Not reset
IEBBnSTC0	Not reset
IEBBnSTC1	Not reset
IEBBnDR	Data written from the CPU is not reset but data received during communication is.

(b) Master request flag (IEBBnMSRQ): Bit 6

- Set/clear condition

  Set: By software
  Clear:

  – Single mode:

  The flag is cleared (to 0) by hardware when master communication is started and when the start interrupt of the master occurs.

  The flag is cleared (to 0) by hardware when a communication error interrupt occurs (when the EBBnISR.IEBBnIEBE bit = 1).

  The flag is cleared (to 0) by hardware when arbitration loss occurs.

  The flag is cleared (to 0) when the IEBBnPW bit is cleared.

  – FIFO mode:

  The flag is cleared (to 0) by hardware after master communication starts, communication is performed without arbitration loss occurring, and the parity bit of the slave address field output by the unit is transmitted.

  The flag is cleared (to 0) by hardware when a communication error interrupt occurs (when the EBBnISR.IEBBnIEBE bit = 1).

  The flag is cleared (to 0) by hardware if arbitration losses consecutively occur the number of times specified by the IEBBnTMS.IEBBnALC2 to IEBBnALC0 bits.

  The flag is cleared (to 0) when the IEBBnPW bit is cleared.

When the IEBBnMSRQ bit is set (to 1), the IEBus controller starts communication on IEBus as the master.

If communication is in progress on IEBus (if the start bit cannot be detected while the start bit is being communicated or if communication is in progress after the start bit has been detected), however, the controller waits until the current frame ends (holds the master request pending), outputs the start bit after the frame has ended, and starts communication as the master.

Cautions

1. Only set the IEBBnMSRQ bit after clearing the IEBBnSTXE bit to 0. After setting the IEBBnSTXE bit to 1, if arbitration loss occurs and the slave is selected, the transmission data prepared for the master might be used as slave transmission data.
2. Reissue master requests in the single mode and FIFO mode as described below.
   - Single mode:
     When arbitration is lost, use software to reissue master requests.
   - FIFO mode:
     When arbitration is lost, use hardware to reissue master requests.

     However, if arbitration losses consecutively occur the number of times specified by the IEBBnTMS.IEBBnALC2 to IEBBnALC0 bits, the request must be reissued by using software.

(c) Broadcast request flag (IEBBnALRQ): Bit 5

- Set/clear condition

  Set: By software
  Clear: By software

Cautions

1. The IEBBnMSRQ bit is cleared (to 0) by hardware, but the IEBBnALRQ bit is not. Therefore, if the next master request is for individual communication, clear the IEBBnALRQ bit (to 0).
2. Be sure to change the value of the IEBBnALRQ bit before setting the IEBBnMSRQ bit (to 1).

(d) Slave transmission enable flag (IEBBnSTXE): Bit 4

- Set/clear condition

  Set: By software
  Clear: By software

Slave transmission is controlled by the value of the slave transmission enable flag, but whether IEBBn performs slave transmission (whether there is an $\overline{\text{ACK}}$ signal response for the control field) is determined by other conditions.

The $\overline{\text{ACK}}$ signal response conditions for the control field are shown below.

Table 30-12 Control field $\overline{\text{ACK}}$ signal response conditions (when the received control data is 0H, 3H, 4H, 5H, 6H, or 7H)

Communication target (IEBBnUSR. IEBBnSRQF bit) Slave specification = 1 No specification = 0	Lock status (IEBBnUSR. IEBBnLCKF bit) Lock = 1 No lock = 0	Master unit judgment (IEBBnPAR register match) Lock request unit = 1 Other = 0	Slave transmission enabled (IEBBnBCR.IEBBnSTXE bit)	Slave reception enabled (IEBBnBCR. IEBBnSRXE bit)	Received Control Data					
					0H	3H	4H	5H	6H	7H
1	0	don't care	0	don't care	A	N	N	N	A	N
			1		A	A	N	N	A	A
	1	0	don't care		A	N	A	A	N	N
		1	0		A	N	A	A	A	N
			1		A	A	A	A	A	A
Other than the above					N					

Note A: Slave transmission is performed. (The $\overline{\text{ACK}}$ signal is returned.)

N: Slave transmission is not performed. (The NACK signal is returned.)

Slave transmission is not performed if the received control data is AH, BH, EH, or FH.

Table 30-13 Control field $\overline{ACK}$ signal response conditions (when the received control data is AH, BH, EH, or FH)

Communication target (IEBBnUSR. IEBBnSRQF bit) Slave specification = 1 No specification = 0	Lock status (IEBBnUSR. IEBBnLCKF bit) Lock = 1 No lock = 0	Master unit judgment (IEBBnPAR register match) Lock request unit = 1 Other = 0	Slave transmission enabled (IEBBnBCR.IEBBnSTXE bit)	Slave reception enabled (IEBBnBCR.IEBBnSRXE bit)	Received control data			
					AH	BH	EH	FH
1	0	don't care	don't care	1	A			
	1	1						
Other than the above					N			

Note A: The $\overline{ACK}$ signal is returned.

N: The NACK signal is returned.

Cautions

1. Set the IEBBnSTXE bit before the control field parity bit is received.
2. When there is a master request, clear the IEBBnSTXE bit (to 0) before setting the IEBBnMSRQ bit (to 1). This is to avoid transmission of the data of the IEBBnDR register that tries master transmission if the controller loses arbitration after master operation and if slave transmission is requested by the master.

(e) Slave reception enable flag (IEBBnSRXE): Bit 3

- Set/clear condition

  Set: By software
  Clear: By software

When the IEBBnSRXE bit = 1 and the reception control data for the communicated control field addressed to the unit is AH, BH, EH, or FH (or when the lock status is specified and the master unit address of the communication matches the address for which a lock was requested), the ACK signal is returned for the control field, and a slave reception operation is performed.

When the IEBBnSRXE bit = 0 and the reception control data for the communicated control field addressed to the unit is AH, BH, EH, or FH, the NACK signal is returned for the control field, and no slave reception operation is performed.

Cautions

1. Set the IEBBnSRXE bit before the control field parity bit is received.
2. The IEBBnSRXE bit is used to enable or disable slave reception for both individual and broadcast communication. For individual communication, a NACK signal can be returned for the control field to end communication by clearing the IEBBnSRXE bit to 0 (thereby prohibiting slave reception), but, for broadcast communication, although communication cannot be ended by clearing this bit because no $\overline{\text{ACK}}$/NACK signal is transmitted, no data interrupt occurs because IEBBn does not respond to the broadcast communication.

(2) IEBBnPSR - IEBBn power save register

The IEBBnPSR register is used to operate and stop the IEBBn operation clock and to control the communication mode.

Access This register can be read or written in 8-bit or 1-bit units.

Address <IEBBn_base> + 0004H

Initial value 00H

Cautions

1. The IEBBnPSR register can only be set up when the IEBBnBCR.IEBBnPW bit = 0. Do not set up the register when this bit = 1. If an attempt is made to set up the register when the IEBBnPW bit = 1, the value is ignored.
2. To use IEBBn, first set the IEBBnCLKE bit (to 1) and enable the operation clock.

   To start the bus operation, specify the settings below.

   - When communication has started

   1. Set up the IEBBnCKS register.
   2. Set the IEBBnCLKE bit (to 1). (The operation clock operates.)

      Set the IEBBnCMD bit to 0 or 1 to specify the communication mode.
   3. Set up registers such as IEBBnUAR, IEBBnSAR, IEBn0TCD, IEBBnTDL, and IEBBnDR depending on the type of communication.
   4. Set the IEBBnBCR.IEBBnPW bit (to 1) to start communication.

   - When communication is stopped

   1. Clear the IEBBnPW bit to 0.
   2. Clear the IEBBnCLKE bit (to 0). (The operation clock stops.)

7	6	5	4	3	2	1	0
IEBBn CLKE	IEBBn CMD	0	0	0	0	0	0
R/W	R/W	R	R	R	R	R	R

Table 30-14 IEBBnPSR register contents

Bit position	Bit name	Function
7	IEBBnCLKE	Operation clock enable flag 0: Stop the operation clock. (This makes it possible to reduce the power consumed by IEBBn.) Initialize the prescaler and baud rate generator. 1: Enable the operation clock. The operation clock starts operating one clock cycle after the IEBBnCLKE bit is set (to 1). (For details, see *Figure 30-2 "Starting and stopping the operation clock".*) Similarly, the operation clock stops operating one clock cycle after the IEBBnCLKE bit is cleared (to 0). (For details, see *Figure 30-2 "Starting and stopping the operation clock".*)
6	IEBBnCMD	IEBBn communication mode setting flag 0: Specify mode 1 as the operation mode. 1: Specify mode 2 as the operation mode.

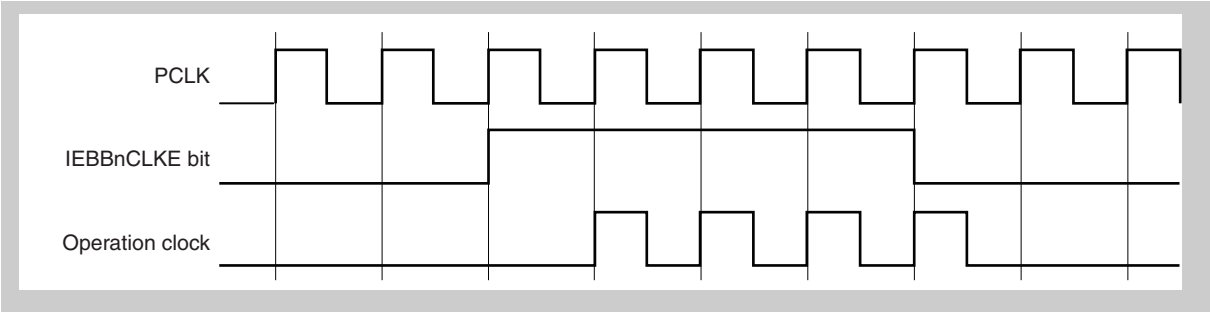


Figure 30-2 Starting and stopping the operation clock

(3) IEBBnUAR - IEBBn unit address register

The IEBBnUAR register is used to specify the unit address of the IEBus unit. This register must always be set before starting communication.

Specify the unit address (12 bits) for bits 11 to 0.

Access This register can be read or written in 16-bit units.

Address <IEBBn_base> + 0008H

Initial value 0000H

Cautions

1. The IEBBnUAR register can only be set up when the IEBBnBCR.IEBBnPW bit = 0. Do not set up the register when this bit = 1. If an attempt is made to set up the register when the IEBBnPW bit = 1, the value is ignored.
2. Writing to this register in 8-bit units is prohibited.

15	14	13	12	11	10	9	8
0	0	0	0				
R	R	R	R	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

(4) IEBBnSAR - IEBBn slave address register

The IEBBnSAR register is used to specify the address of the communication-partner slave unit during master communication.

During a master request, the value of this register is transmitted as the slave address field data.

Specify the slave address (12 bits) for bits 11 to 0.

Access This register can be read or written in 16-bit units.

Address <IEBBn_base> + 000CH

Initial value 0000H

Cautions

1. When the IEBBnSAR register is overwritten during communication (while the IEBBnBCR.IEBBnPW bit = 1), communication might not be correctly performed. Therefore, overwriting is prohibited from when a master request is issued until the communication or frame completion timing.

   Note that overwriting is enabled at the following times:

   - When the IEBBnPW bit = 0
   - From when the IEBBnPW bit is set to 1 until the first master request (when the IEBBnMSRQ bit = 1)
   - From the communication or frame completion timing (assuming the IEBBnPW bit = 1 and the IEBBnMSRQ bit = 0) until the next master request (when the IEBBnMSRQ bit = 1)

2. Writing to this register in 8-bit units is prohibited.

15	14	13	12	11	10	9	8
0	0	0	0				
R	R	R	R	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

(5) IEBBnPAR - IEBBn partner address register

The IEBBnPAR register is used to store the reception master address in the master address field.

When operating the unit is enabled (when the IEBBnBCR.IEBBnPW bit = 1), the received master address is stored using the master address field regardless of whether the unit master is operating or the slave is operating.

- Storage in the single mode

  Upon completion of the parity period for the master address field, this is only performed if the parity value is normal and the unit is in the non-lock status.

- Storage in the FIFO mode

  If reading the data received during the previous communication has finished (if the IEBBnBSR.IEBBnRFLF bit = 0 and the IEBBnBSR.IEBBnSRFP4 to IEBBnSRFP0 bits = 00000), upon completion of the parity period for the master address field, storage is only performed if the parity value is normal and the unit is in the non-lock status. If reading the received data has not finished, the IEBBnPAR register is not updated until it finishes.

When there is a unit lock, because the address of the unit that requested the lock (the lock master) is retained, the IEBBnPAR register is not updated.

- Lock address transmission request reception in the single mode

  If a lock address transmission request is received from the master as a status transmission request, when the received control data receives the lock address (higher four bit) read request (5H), the value of the IEBBnPAR register is read by using software, and then the data in bits 15 to 8 of the IEBBnPAR register is written to the IEBBnDR register.

  In addition, if a lock address (lower 8 bits) read request (4H) is received, the value of the IEBBnPAR register is read by using software, and then the data in bits 7 to 0 of the IEBBnPAR register is written to the IEBBnDR register.

- Lock address transmission request reception in the FIFO mode

  If a lock address transmission request is received from the master as a status transmission request, the data in the IEBBnPAR register is automatically transmitted to the data field by using hardware.

Specify the partner address (12 bits) for bits 11 to 0.

Access This register is read-only, in 16-bit units.

Address <IEBBn_base> + 0010H

Initial value 0000H

This register is reset when the IEBBnBCR.IEBBnPW bit is overwritten.

15	14	13	12	11	10	9	8
0	0	0	0				
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
R	R	R	R	R	R	R	R

(6) IEBBnRSA - IEBBn reception slave address register

The IEBBnRSA register is used to store the slave address value received using the slave address field.

When operating the unit is enabled (when the IEBBnBCR.IEBBnPW bit = 1), the received slave address is stored using the slave address field regardless of whether the unit master is operating or the slave is operating.

- Storage in the single mode

  This is performed upon the completion of the slave address field parity period if the parity value is normal.

- Storage in the FIFO mode

  If reading the data received during the previous communication has finished (if the IEBBnBSR.IEBBnRFLF bit = 0 and the IEBBnBSR.IEBBnSRFP4 to IEBBnSRFP0 bits = 00000), upon completion of the parity period for the slave address field, storage is only performed if the parity value is normal. Until reading the received data finishes, the IEBBnRSA register is not updated.

Specify the slave address (12 bits) for bits 11 to 0.

Access This register is read-only, in 16-bit units.

Address <IEBBn_base> + 0014H

Initial value 0000H

This register is reset when the IEBBnBCR.IEBBnPW bit is overwritten.

15	14	13	12	11	10	9	8
0	0	0	0				
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
R	R	R	R	R	R	R	R

(7) IEBBnCDR - IEBBn control data register

The IEBBnCDR register is used to specify the control data transmitted using the control field.

After writing to the IEBBnCDR register, the IEBBnTCD register is written to.

After reading the IEBBnCDR register, the IEBBnRCD register value is read.

Access This register can be read or written in 8-bit units.

Address <IEBBn_base> + 0018H

Initial value 00H

The read value is reset when the IEBBnBCR.IEBBnPW bit is overwritten.

Caution When issuing a master request, be sure to set up the IEBBnCDR register before starting communication (when the IEBBnBCR.IEBBnMSRQ bit = 0).

Note The IEBBnCDR register consists of a write register and a read register. Therefore, data written to this register cannot be read as is. The data received during IEBus communication can be read.

7	6	5	4	3	2	1	0
0	0	0	0	IEBBn SLCD3	IEBBn SLCD2	IEBBn SLCD1	IEBBn SLCD0
R	R	R	R	R/W	R/W	R/W	R/W

(8) IEBBnTCD - IEBBn transmission control data register

The IEBBnTCD register is used to specify the control data transmitted using the control field.

The value of the lower 4 bits of the value written to the IEBBnTCD register is transmitted as control data by using the control field during master transmission.

Access This register can be read or written in 8-bit units.

Address <IEBBn_base> + 001CH

Initial value 00H

Cautions

1. When issuing a master request, be sure to set up the IEBBnTCD register before starting communication (when the IEBBnBCR.IEBBnMSRQ bit = 0).
2. Do not specify undefined values.
3. During broadcast transmission, specifying slave transmission control data is prohibited.

7	6	5	4	3	2	1	0
0	0	0	0	IEBBn SLTD3	IEBBn SLTD2	IEBBn SLTD1	IEBBn SLTD0
R	R	R	R	R/W	R/W	R/W	R/W

Table 30-15 IEBBnTCD register contents

Bit position	Bit name	Function
3 to 0	IEBBnSLTD [3-0]	Specify the control data transmitted by using the control field.

IEBBn SLTD3	IEBBn SLTD2	IEBBn SLTD1	IEBBn SLTD0	Function
0	0	0	0	Read slave status
0	0	0	1	Undefined
0	0	1	0	Undefined
0	0	1	1	Data reading and locking
0	1	0	0	Lock address reading (lower 8 bits)
0	1	0	1	Lock address reading (higher 4 bits)
0	1	1	0	Slave status reading and unlocking
0	1	1	1	Read data
1	0	0	0	Undefined
1	0	0	1	Undefined
1	0	1	0	Command writing and locking
1	0	1	1	Data writing and locking
1	1	0	0	Undefined
1	1	0	1	Undefined
1	1	1	0	Write command
1	1	1	1	Write data

(9) IEBBnRCD - IEBBn reception control data register

The IEBBnRCD register is used to store the control data received using the control field.

The data received by using the control field is read to the lower 4 bits of the IEBBnRCD register. Data is stored in the IEBBnRCD register upon completion of the control field parity period if the parity value is normal.

- Storage in the single mode

  When a status transmission request is received, the user performs each process (settings for the transmission data of the IEBBnSSR register or the IEBBnPAR register) according to the value of the lower 4 bits of the IEBBnRCD register read value.

- Storage in the FIFO mode

  When a status transmission request is received, the hardware automatically performs the status transmission processing (settings for the transmission data of the IEBBnSSR register or the IEBBnPAR register).

Because it is necessary to judge whether the received data is a command or data, be sure to read the value of this register upon the completion of communication.

In the FIFO mode, if reading the data received during the previous communication has finished (if the IEBBnBSR.IEBBnRFLF bit = 0 and the IEBBnBSR.IEBBnSRFP4-SRFP0 bit = 00000), upon completion of the parity period for the control data field, storage is performed if the parity value is normal. Until reading the received data finishes, the IEBBnRCD register is not updated.

Access This register is read-only, in 8-bit units.

Address <IEBBn_base> + 0020H

Initial value 00H

This register is reset when the IEBBnBCR.IEBBnPW bit is overwritten.

7	6	5	4	3	2	1	0
0	0	0	0	IEBBn SLRD3	IEBBn SLRD2	IEBBn SLRD1	IEBBn SLRD0
R	R	R	R	R	R	R	R

Table 30-16 IEBBnRCD register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>3 to 0</td><td>IEBBnSLRD[3-0]</td><td>Specify the control data received by using the control field.
<table>
<tr><th>IEBBn SLRD3</th><th>IEBBn SLRD2</th><th>IEBBn SLRD1</th><th>IEBBn SLRD0</th><th>Function</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Read slave status</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>Undefined</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>Undefined</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>Data reading and locking</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>Lock address reading (lower 8 bits)</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>Lock address reading (higher 4 bits)</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>Slave status reading and unlocking</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>Read data</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>Undefined</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>Undefined</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>Command writing and locking</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>Data writing and locking</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>Undefined</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>Undefined</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>Write command</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>Write data</td></tr>
</table></td></tr>
</table>

(10) IEBBnDLR - IEBBn message length register

The IEBBnDLR register is used to specify the message length data transmitted using the message length field.

After writing to the IEBBnDLR register, the IEBBnTDL register is written to.

After reading the IEBBnDLR register, the IEBBnRDL register value is read.

Access This register can be read or written in 8-bit units.

Address <IEBBn_base> + 0024H

Initial value 01H

The read value is reset when the IEBBnBCR.IEBBnPW bit is overwritten.

Caution When issuing a master request, be sure to set up the IEBBnDLR register before starting communication (when the IEBBnBCR.IEBBnMSRQ bit = 0).

Note The IEBBnDLR register consists of a write register and a read register. Therefore, data written to this register cannot be read as is. The data received during IEBus communication can be read.

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

(11) IEBBnTDL - IEBBn transmission message length register

The IEBBnTDL register is used to specify the message length data transmitted using the message length field.

The value written to the IEBBnTDL register is transmitted as message length data by using the message length field if the unit is the transmission unit (master transmission, slave transmission).

However, when a status transmission request is received, 0H is transmitted as the message length data regardless of the IEBBnTDL register setting.

Access This register can be read or written in 8-bit units.

Address <IEBBn_base> + 0028H

Initial value 01H

Cautions

1. Be sure to set up the IEBBnTDL register before starting communication (when the IEBBnBCR.IEBBnMSRQ bit = 0).
2. The maximum number of bytes that can be transferred per frame is determined according to the communication mode. For example, when transferring 48 bytes in mode 1, perform communication by dividing the data amoung multiple frames. In this case, when performing the second communication, use the IEBBnSCR register to check the number of data bytes transmitted during the first communication, subtract the number of bytes that were successfully transmitted from the number of bytes you want to transmit, and then specify the result for the IEBBnTDL register. Write the next data to the IEBBnDR register at the same time, and then issue a master request.

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-17 IEBBnTDL register contents

Bit								Setting	Number of remaining communication data bytes
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	01H	1 byte
0	0	0	0	0	0	1	0	02H	2 bytes
...	...	...	...	...	...	...	...	...	...
0	0	0	1	0	1	0	0	20H	32 bytes
...	...	...	...	...	...	...	...	...	...
1	1	1	1	1	1	1	1	FFH	255 bytes
0	0	0	0	0	0	0	0	00H	256 bytes

(12) IEBBnRDL - IEBBn reception message length register

The IEBBnRDL register is used to specify the message length data received using the message length field.

The IEBBnRDL register read value is the data received using the message length field. Data is stored in the IEBBnRDL register upon completion of the message field parity period if the parity value is normal.

- Storage in the FIFO mode

  If reading the data received during the previous communication has finished (IEBBnBSR.IEBBnRFLF bit = 0 and the IEBBnBSR.IEBBnSRFP4-SRFP0 bit = 00000), upon completion of the parity period for the message length field, storage is performed if the parity value is normal. Until reading the received data finishes, the IEBBnRDL register is not updated.

Access This register is read-only, in 8-bit units.

Address <IEBBn_base> + 002CH

Initial value 01H

This register is reset when the IEBBnBCR.IEBBnPW bit is overwritten.

7	6	5	4	3	2	1	0
R	R	R	R	R	R	R	R

Table 30-18 IEBBnRDL register contents

Bit								Setting	Number of remaining communication data bytes
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	01H	1 byte
0	0	0	0	0	0	1	0	02H	2 bytes
...	...	...	...	...	...	...	...	...	...
0	0	0	1	0	1	0	0	20H	32 bytes
...	...	...	...	...	...	...	...	...	...
1	1	1	1	1	1	1	1	FFH	255 bytes
0	0	0	0	0	0	0	0	00H	256 bytes

(13) IEBBnCKS - IEBBn clock selection register

The IEBBnCKS register is used to control the clock selection of the IEBus controller.

Access This register can be read or written in 8-bit units.

Address <IEBBn_base> + 0030H

Initial value 17H

Caution The IEBBnCKS register can only be set up when the IEBBnBCR.IEBBnPW bit = 0 and the IEBBnPSR.IEBBnCLKE bit = 0. Do not set up the register when this bit = 1. If an attempt is made to set up the register when the IEBBnPW bit = 1, the value is ignored.

7	6	5	4	3	2	1	0
0	0	0	IEBBn PRS	0	IEBBn BRS2	IEBBn BRS1	IEBBn BRS0
R	R	R	R/W	R	R/W	R/W	R/W

Table 30-19 IEBBnCKS register contents

Bit position	Bit name	Function
4	IEBBnPRS	Specify the prescaler output (PRSOUT). 0: PCLK 1: PCLK/2 **Caution** The conditions under which the prescaler is initialized are as follows: • When the IEBBnPRS bit is overwritten • When the IEBBnBCR.IEBBnPW bit = 0 and the IEBBnPSR.IEBBnCLKE bit = 1
2 to 0	IEBBnBRS [2-0]	Specify the operation clock output (MCK). **Caution** The conditions under which the baud rate generator is initialized are as follows: • When the IEBBnBRS2 to IEBBnBRS0 bits are overwritten • When the IEBBnBCR.IEBBnPW bit = 0 and the IEBBnPSR.IEBBnCLKE bit = 1

IEBBnBRS2	IEBBnBRS1	IEBBnBRS0	Operation clock output (MCK)
0	0	0	PRSOUT/1
0	0	1	PRSOUT/1
0	1	0	PRSOUT/2
0	1	1	PRSOUT/3
1	0	0	PRSOUT/4
1	0	1	PRSOUT/5
1	1	0	PRSOUT/6
1	1	1	PRSOUT/7

Table 30-20 Input clock specification example

PCLK	IEBBnPRS	IEBBnBRS2	IEBBnBRS1	IEBBnBRS0	Specified value
80 MHz	1	1	0	1	15H
64 MHz	1	1	0	0	14H
48 MHz	1	0	1	1	13H
40 MHz	0	1	0	1	05H
32 MHz	0	1	0	0	04H

Caution The IEBus controller is designed to operate at 8 MHz.

(14) IEBBnTMS - IEBBn transfer mode setting register

The IEBBnTMS register is used to control the IEBus controller communication operations.

Access This register can be read or written in 8-bit units.

Address <IEBBn_base> + 0034H

Initial value 01H

Caution The IEBBnTMS register can only be set up when the IEBBnBCR.IEBBnPW bit = 0. Do not set up the register when this bit = 1. If an attempt is made to set up the register when the IEBBnPW bit = 1, the value is ignored.

7	6	5	4	3	2	1	0
IEBBn FMDE	IEBBn SLRI1	IEBBn SLRI0	IEBBn SLTI1	IEBBn SLTI0	IEBBn ALC2	IEBBn ALC1	IEBBn ALC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-21 IEBBnTMS register contents (1/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7</td><td>IEBBnFMDE</td><td>Specify whether to enable or disable FIFO mode operation.
0: Disable FIFO mode operation (single mode).
1: Enable FIFO mode operation.</td></tr>
<tr><td>6, 5</td><td>IEBBnSLRI [1, 0]</td><td>Specify the IEBBTV occurrence timing during FIFO mode reception.
<table>
<tr><th>IEBBnSLRI1</th><th>IEBBnSLRI0</th><th>IEBBTV occurrence timing</th></tr>
<tr><td>0</td><td>0</td><td>When the reception data that has not been read from the reception FIFO buffer reaches 32 bytes</td></tr>
<tr><td>0</td><td>1</td><td>When the reception data that has not been read from the reception FIFO buffer reaches 24 bytes</td></tr>
<tr><td>1</td><td>0</td><td>When the reception data that has not been read from the reception FIFO buffer reaches 16 bytes</td></tr>
<tr><td>1</td><td>1</td><td>When the reception data that has not been read from the reception FIFO buffer reaches 8 bytes</td></tr>
</table></td></tr>
<tr><td>4, 3</td><td>IEBBnSLTI [1, 0]</td><td>Specify the IEBBTD occurrence timing during FIFO mode transmission.
<table>
<tr><th>IEBBnSLTI1</th><th>IEBBnSLTI0</th><th>IEBBTD occurrence timing</th></tr>
<tr><td>0</td><td>0</td><td>When the transmission FIFO buffer becomes empty</td></tr>
<tr><td>0</td><td>1</td><td>When the untransmitted data remaining in the transmission FIFO buffer reaches 2 bytes</td></tr>
<tr><td>1</td><td>0</td><td>When the untransmitted data remaining in the transmission FIFO buffer reaches 4 bytes</td></tr>
<tr><td>1</td><td>1</td><td>When the untransmitted data remaining in the transmission FIFO buffer reaches 8 bytes</td></tr>
</table></td></tr>
</table>

Table 30-21 IEBBnTMS register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>2 to 0</td><td>IEBBnALC [2-0]</td><td>Specify the arbitration loss number.
This is only valid in the FIFO mode. The settings of these bits are invalid in the single mode.
When arbitration is lost and the counter is set to 0H, the IEBBnBCR.IEBBnMSRQ bit is not retained.
The settings of the IEBBnALC2 to IEBBnALC0 bits are always retained. The settings are not changed each time arbitration is lost.
The arbitration loss counter is decremented separately from the IEBBnALC2 to IEBBnALC0 bits.
Overwriting the IEBBnALC2 to IEBBnALC0 bits while the arbitration loss counter is counting does not affect the counter. (The values of the bits are specified for the arbitration loss counter when the IEBBnMSRQ bit = 1.)
<table>
<tr><th>IEBBnALC2</th><th>IEBBnALC1</th><th>IEBBnALC0</th><th>Maximum arbitration loss count</th></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>3</td></tr>
<tr><td colspan="3">Other than the above</td><td>Setting prohibited</td></tr>
</table>
</td></tr>
</table>

(a) FIFO operation enable bit (IEBBnFMDE): Bit 7

Differences between operation in the single mode and FIFO mode are shown below.

Table 30-22 Differences between operation in the single mode and FIFO mode

Transfer mode	Arbitration loss master request flag (IEBBnBCR.IEBBnMSRQ)	Arbitration loss error interrupt signal (IEBBTERR)	Support for slave status requests and lock address requests
Single mode	Clear	• The signal is not output. • The IEBBnESR.IEBBnABTE bit is fixed to 0.	During the slave status interrupt servicing, the value of the IEBBnSSR or IEBBnPAR register is written to the IEBBnDR register.
FIFO mode	The flag value is retained until arbitration is lost the number of times specified for the IEBBnALC2 to IEBBnALC0 bits. (The flag is cleared unless an error occurs during communication between third parties.)	• The signal is output if arbitration is lost the specified number of times. • The IEBBnESR.IEBBnABTE bit is simultaneously set (to 1).	The value of the IEBBnSSR or IEBBnPAR register is automatically sent by hardware by using the data field.

(b) FIFO mode reception IEBBTV occurrence timing specification bits (IEBBnSLRI1, IEBBnSLRI0): Bits 6, 5

In communication mode 1, the IEBBnSLRI1 and IEBBnSLRI0 bits are cleared to 00.

In communication mode 2, because data that exceeds 32 bytes is received, data must be read during reception. The occurrence timing of the data interrupt (IEBBTV) that requests that received data be read is specified by the IEBBnSLRI1 and IEBBnSLRI0 bits.

The IEBBTV interrupt occurs when received data is stored in the FIFO buffer and the number of unread bytes of data reaches the value specified for the IEBBnSLRI1 and IEBBnSLRI0 bits.

For the data exceeding the number of bytes specified for the IEBBnSLRI1 and IEBBnSLRI0 bits, the data is read, and no interrupt occurs even if the number of unread bytes of received data exceeds the value specified for the IEBBnSLRI1 and IEBBnSLRI0 bits.

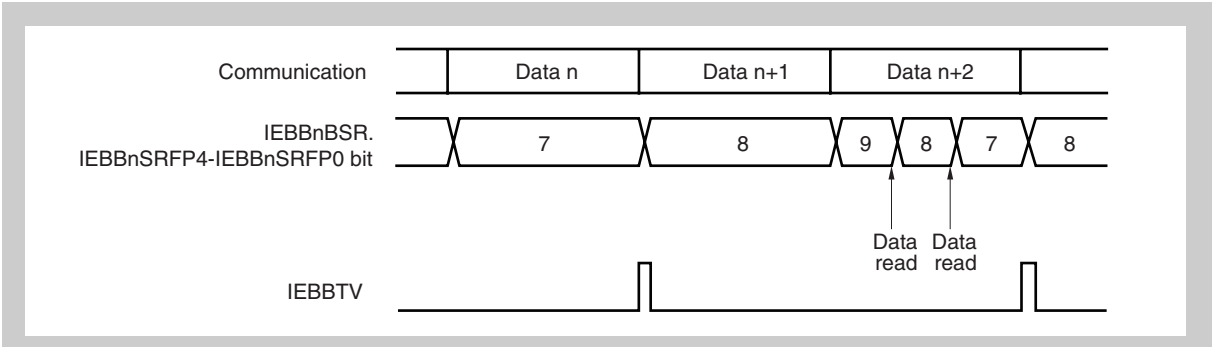


Figure 30-3 Example of operation in communication mode 2: When the IEBBnSLRI1 and IEBBnSLRI0 bits = 11

(c) FIFO mode transmission IEBBTB occurrence timing specification bits (IEBBnSLTI1, IEBBnSLTI0): Bits 4, 3

In communication mode 1, clear the IEBBnSLTI1 and IEBBnSLTI0 bits to 00.

In communication mode 2, because data that exceeds 32 bytes is transmitted, transmission data must be written during reception. The occurrence timing of the data interrupt (IEBBTD) that requests that transmission data be written is specified by the IEBBnSLTI1 and IEBBnSLTI0 bits.

When less than 32 bytes of data are written to the FIFO buffer, IEBBTD occurs for the remaining number of bytes of data to be transmitted.

The IEBBTD interrupt occurs when transmission data is transferred from the FIFO buffer to the shift register and the number of bytes of data that have not been transmitted reaches the value specified for the IEBBnSLTI1 and IEBBnSLTI0 bits.

For data less than the number of bytes specified for the IEBBnSLTI1 and IEBBnSLTI0 bits, the data is written, and no interrupt occurs even if the number of bytes of data that have not been transmitted reaches the value specified for the IEBBnSLTI1 and IEBBnSLTI0 bits.

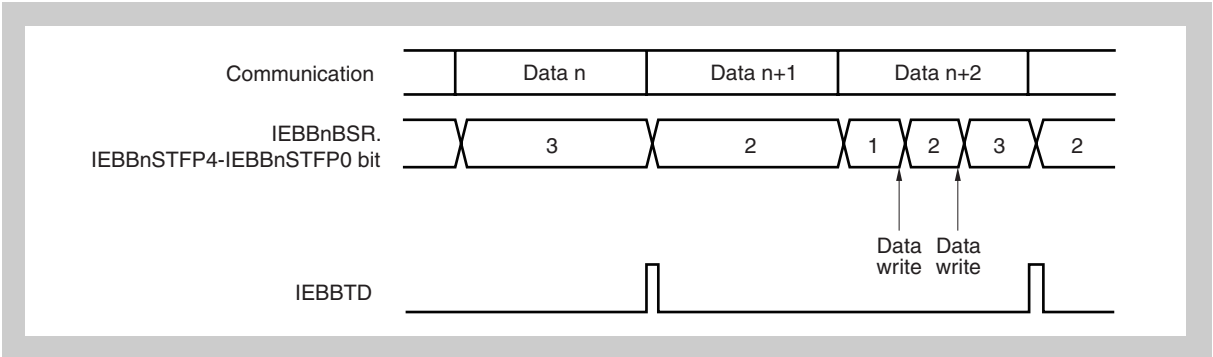


Figure 30-4 Example of operation in communication mode 2: When the IEBBnSLTI1 and IEBBnSLTI0 bits = 01

(d) Arbitration loss count specification bits (IEBBnALC2 to IEBBnALC0): Bits 2 to 0

This is only valid in the FIFO mode. The settings of these bits are invalid in the single mode.
When arbitration is lost and the counter is set to 0H, the IEBBnBCR.IEBBnMSRQ bit is not retained.

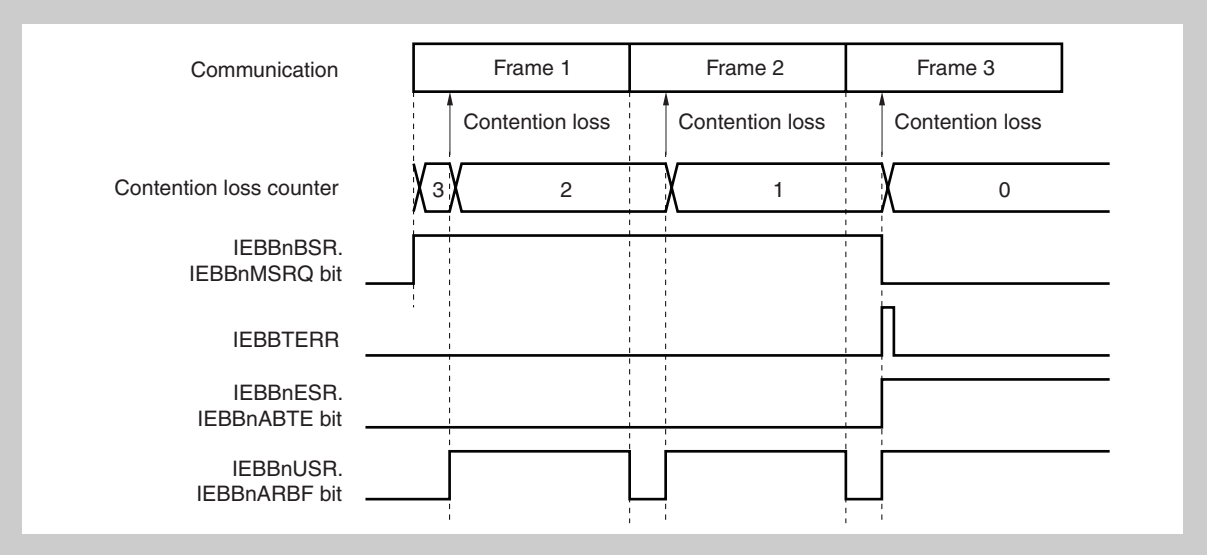


Figure 30-5 Example of operation in the FIFO mode: When the IEBBnALC2 to IEBBnALC0 bits = 011

(15) IEBBnPCR - IEBBn pointer clear register

The IEBBnPCR register is the trigger bit register for clearing the FIFO buffer pointer.

Access This register is write-only, in 8-bit units.

Address <IEBBn_base> + 0038H

Initial value 00H

7	6	5	4	3	2	1	0
IEBBn CRPT	IEBBn CTPT	0	0	0	0	0	0
W	W	W	W	W	W	W	W

Table 30-23 IEBBnPCR register contents

Bit position	Bit name	Function
7	IEBBnCRPT	Clear trigger bit for the store pointer and read pointer of the reception FIFO buffer 0: No operation 1: Clear the store pointer and read pointer of the reception FIFO buffer. The bit value can only be changed by setting the bit (to 1). Attempting to clear the bit (to 0) does not change the bit value. When the bit is read, 0 is always returned. **Caution** During reception or before reading received data, if the IEBBnCRPT bit is set to 1, the received data cannot be read. Except when discarding received data, only write 1 to the IEBBnCRPT bit after receiving and then reading data.
6	IEBBnCTPT	Clear the trigger bit for the write pointer and load pointer of the transmission FIFO buffer 0: No operation 1: Clear the write pointer and load pointer of the transmission FIFO buffer. The bit value can only be changed by setting the bit (to 1). Attempting to clear the bit (to 0) does not change the bit value. When the bit is read, 0 is always returned. **Cautions 1.** If the IEBBnCTPT bit = 1 while specifying data for the FIFO buffer or during transmission, the specified data cannot be transmitted. Except when discarding transmission data, only write 1 to the IEBBnCTPT bit after transmitting the data written to the FIFO buffer. **2.** To discard transmission data, set the IEBBnCTPT bit to 1 during transmission. Because the data is lost in this case, transmission is not performed and an underrun error occurs.

(16) IEBBnBSR - IEBBn buffer status register

The IEBBnBSR register indicates the FIFO buffer status.

Access This register is read only, in 16-bit units.

Address <IEBBn_base> + 003CH

Initial value 0000H

This register is reset when 0 is written to the IEBBnBCR.IEBBnPW bit.

15	14	13	12	11	10	9	8
IEBBn RFLF	IEBBn FOVR	0	IEBBn SRFP4	IEBBn SRFP3	IEBBn SRFP2	IEBBn SRFP1	IEBBn SRFP0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
IEBBn TFLF	IEBBn FOVW	0	IEBBn STFP4	IEBBn STFP3	IEBBn STFP2	IEBBn STFP1	IEBBn STFP0
R	R	R	R	R	R	R	R

Table 30-24 IEBBnBSR register contents (1/2)

Bit position	Bit name	Function
15	IEBBnRFLF	Reception FIFO buffer full status flag 0: There are 31 bytes of data or less that have not been read in the reception FIFO buffer. 1: There are 32 bytes of data that have not been read in the reception FIFO buffer. This flag indicates that there are 32 bytes of data that have not been read in the reception FIFO buffer and that the buffer is full.
14	IEBBnFOVR	This flag indicates whether reception FIFO buffer over-reading has occurred. 0: Reception FIFO buffer over-reading has not occurred. 1: Reception FIFO buffer over-reading has occurred.
12 to 8	IEBBnSRFP[4-0]	This flag indicates the number of data bytes that have not been read remaining in the reception FIFO buffer. The (store pointer - read pointer) value can be read. However, the following values are used when IEBBnSRFP4 to IEBBnSRFP0 = 00000. • When the IEBBnRFLF bit = 0 and the IEBBnSRFP4 to IEBBnSRFP0 bits = 00000 Number of remaining data bytes that have not been read = 0 bytes • When the IEBBnRFLF bit = 1, and the IEBBnSRFP4 to IEBBnSRFP0 bits = 00000 Number of remaining data bytes that have not been read = 32 bytes
7	IEBBnTFLF	Transmission FIFO buffer full status flag 0: There are 31 bytes of data or less that have not been transferred in the transmission FIFO buffer. 1: There are 32 bytes of data that have not been transferred in the transmission FIFO buffer. This flag indicates that there are 32 bytes of data that have not been transferred in the transmission FIFO buffer and that the buffer is full.
6	IEBBnFOVW	Transmission FIFO buffer overwrite flag 0: No transmission FIFO buffer overwrite has occurred. 1: A transmission FIFO buffer overwrite has occurred.

Table 30-24 IEBBnBSR register contents (2/2)

Bit position	Bit name	Function
4 to 0	IEBBnSTFP[4-0]	This flag indicates the number of data bytes that have not been transferred remaining in the transmission FIFO buffer. The (write pointer - load pointer) value can be read. However, the following values are used when the IEBBnSTFP4 to STFPSTFP0 bits = 00000. • When the IEBBnTFLF bit = 0 and the IEBBnSTFP4 to IEBBnSTFP0 bits = 00000 Number of remaining data bytes that have not been transferred = 0 bytes • When the IEBBnTFLF bit = 1 and the IEBBnSTFP4 to IEBBnSTFP0 bits = 00000 Number of remaining data bytes that have not been transferred = 32 bytes

(a) Reception FIFO buffer over-read indicating flag (IEBBnFOVR): Bit 14

- Set/clear condition

  Set:

  – Single mode:

    The IEBBnFOVR bit is not set (to 1).

  – FIFO mode:

    When an over-read (a read of the IEBBnDR register while the store pointer = the read pointer) occurs for the reception FIFO buffer

  Clear:

  – Single mode or FIFO mode:

    By software (The flag is cleared when 1 is written to the IEBBnPCR.IEBBnCRPT bit.)

After reading the reception FIFO buffer, detect whether an over-read occurred by reading the IEBBnFOVR bit. (The data read during the over-read is the last bytes read during multiple operations.)

Even if an over-read is detected, because the data cannot be read again, perform software processing such as requesting retransmission. The IEBBnFOVR bit is cleared by writing 1 to the IEBBnPCR.IEBBnCRPT bit. (If an over-read occurred, the FIFO buffer pointers have already been cleared.)

(b) Transmission FIFO buffer overwrite flag (IEBBnFOVW): Bit 6

- Set/clear condition

  Set:

  – Single mode:

    The IEBBnFOVW bit is not set (to 1).

  – FIFO mode:

    When there are 32 bytes of that have not been transmitted in the transmission FIFO buffer and a 33rd byte of data is written

  Clear:

  – Single mode or FIFO mode:

    By software (The flag is cleared when 1 is written to the IEBBnPCR.IEBBnCTPT bit.)

After writing the required data to the transmission FIFO buffer, read the IEBBnFOVW bit before setting the master request flag. (IEBBnBCR.IEBBnMSRQ) (to 1) to detect whether an overwrite has occurred. If an overwrite is detected, clear the FIFO buffer pointers by writing 1 to the IEBBnPCR.IEBBnCTPT bit and then specify the data again.

(17) IEBBnSSR - IEBBn slave status register

The IEBBnSSR register indicates the communication status of the slave unit.

When a slave status transmission request interrupt is received form the master and the received control data is 0H or 6H, the IEBBnSSR register value is automatically written to the IEBBnDR register, and the slave status is transmitted.

In addition, when transmitting the slave status, because 01H is automatically transmitted as the message length, the IEBBnTDL register does not have to be set up.

Because bits 7 and 6 indicate the highest mode supported by the unit, they are fixed to 10 (which indicates mode 2).

Access This register is read-only, in 8-bit or 1-bit units.

Address <IEBBn_base> + 0040H

Initial value 81H

The IEBBnSSLF and IEBBnSTLF bits are reset by writing 0 to the IEBBnBCR.IEBBnPW bit.

The IEBBnSRXF and IEBBnSTXF bits are reset when the value of the IEBBnPW bit is overwritten with a different value.

7	6	5	4	3	2	1	0
1	0	0	IEBBn SSLF	0	IEBBn STLF	IEBBn SRXF	IEBBn STXF
R	R	R	R	R	R	R	R

Table 30-25 IEBBnSSR register contents (1/2)

Bit position	Bit name	Function
4	IEBBnSSLF	Slave transmission status flag 0: Slave transmission is stopped. 1: Slave transmission is enabled.
2	IEBBnSTLF	Lock status flag 0: Unlocked 1: Locked
1	IEBBnSRXF	IEBBnDR register or FIFO buffer reception status flag In the single mode 0: Received data has not been stored in the IEBBnDR register. 1: Received data has been stored in the IEBBnDR register. In the FIFO mode 0: Received data has not been stored in the FIFO buffer. 1: Received data has been stored in the FIFO buffer.

Table 30-25 IEBBnSSR register contents (2/2)

Bit position	Bit name	Function
0	IEBBnSTXF	IEBBnDR register or FIFO buffer transmission status flag • When communication is not being performed 1: The flag is always this value. • Master In the single mode (transmission) 0: The data specified for the IEBBnDR register has been transferred to the transmission shift register, and the next transmission data has not been written to the IEBBnDR register. 1: Transmission data remains in the IEBBnDR register. (This is the status up until the contents of the IEBBnDR register are transferred to the transmission shift register.) In the single mode (reception) 1: The flag is always this value. In the FIFO mode (transmission) 0: The number of bytes of data written to the FIFO buffer have been transferred from the FIFO buffer to the transmission shift register, and the next transfer data has not been written to the FIFO buffer. 1: This is the status from when communication starts until data is transferred to the transmission shift register and the number of bytes of data written to the FIFO buffer disappear. In the FIFO mode (reception) 1: The flag is always this value. • Slave In the single mode (transmission) 0: The data specified for the IEBBnDR register has been transferred to the transmission shift register, and the next transmission data has not been written to the IEBBnDR register. 1: This is the status from when communication starts until the first transmission data is transferred from the IEBBnDR register to the transmission shift register. This is also the status after writing data to the IEBBnDR register until the data is transferred to the transmission shift register. In the single mode (reception) 1: The flag is always this value. In the FIFO mode (transmission) 0: The number of bytes of data written to the FIFO buffer have been transferred from the FIFO buffer to the transmission shift register, and the next transfer data has not been written to the FIFO buffer. 1: This is the status from when communication starts until data is transferred to the transmission shift register and the number of bytes of data written to the FIFO buffer disappear. In the FIFO mode (reception) 1: The flag is always this value. • Third party 1: The flag is always this value.

(a) Slave transmission status flag (IEBBnSSLF): Bit 4

The value of the slave transmission enable flag (the IEBBnBCR.IEBBnSTXE bit) is applied as is.

(b) Lock status flag (IEBBnSTLF): Bit 2

The value of the lock status flag (the IEBBnUSR.IEBBnLCKF bit) is applied as is.

(c) IEBBnDR register or FIFO buffer reception status flag (IEBBnSRXF): Bit 1

- Set/clear condition

  Set:

  – Single mode:

    When received data is stored in the IEBBnDR register

  – FIFO mode:

    When received data is stored in the FIFO buffer

  Clear:

  – Single mode:

    When the IEBBnDR register contents are read

  – FIFO mode:

    When all the received data stored in the FIFO buffer has been read

In the single mode, when the IEBBnSRXF bit is set (to 1), the data interrupt IEBBTD occurs.

In the single mode, when IEBBTD occurs on data reception, the IEBBnDR register must be read before the next data is received. During broadcast communication, an overrun error occurs if the IEBBnDR register is not read regardless of whether IEBBTD has occurred. For details, see the description of the IEBBnOVRE bit in *30.3.2 (20) "IEBBnESR - IEBBn error status register"*.

In the FIFO mode, even if the IEBBnSRXF bit is set (to 1), the interrupt signal IEBBTV is not generated. For details about the IEBBTV generation timing, see the description of the IEBBnSLRI1, IEBBnSLRI0 bits in *30.3.2 (14) "IEBBnTMS - IEBBn transfer mode setting register"*.

(d) IEBBnDR register or FIFO buffer transmission status flag (IEBBnSTXF): Bit 0

- Set/clear condition

  Set:

  – When communication finishes

  – Single mode:

    When the IEBBnDR register is written to

  – FIFO mode:

    When transmission data is written to the FIFO buffer

  Clear:

  – Single mode:

    When the contents of the IEBBnDR register are written to the transmission shift register

  – FIFO mode:

    When the number of bytes of data written to the FIFO buffer are transferred from the FIFO buffer to the transmission shift register

In the single mode, when the IEBBnSTXF bit is cleared, the data interrupt IEBBTD occurs.

In the single mode, when IEBBTD occurs on data transmission, the next transmission data must be written to the IEBBnDR register.

Regardless of whether IEBBTD occurs, an underrun error occurs if IEBBnDR is not written to. For details, see the description of the IEBBnUNRE bit in *30.3.2 (20) "IEBBnESR - IEBBn error status register"*.

In the FIFO mode, even if the IEBBnSRXF bit is set (to 1), the interrupt signal IEBBTD is not generated. For details about the IEBBTD generation timing, see the description of the IEBBnSLTI1, IEBBnSLTI0 bits in *30.3.2 (14) "IEBBnTMS - IEBBn transfer mode setting register"*.

(18) IEBBnUSR - IEBBn unit status register

The IEBBnUSR register indicates the unit status.

Access This register is read-only, in 8-bit or 1-bit units.

Address <IEBBn_base> + 0044H

Initial value 00H

This register is reset when the value of the IEBBnPW bit is overwritten with a different value.

7	6	5	4	3	2	1	0
0	IEBBn SRQF	IEBBn ARBF	IEBBn ALTF	IEBBn ACKF	IEBBn LCKF	0	0
R	R	R	R	R	R	R	R

Table 30-26 IEBBnUSR register contents

Bit position	Bit name	Function
6	IEBBnSRQF	Slave request flag for the unit 0: There are no slave requests. 1: There is a slave request.
5	IEBBnARBF	Arbitration result flag 0: Arbitration loss did not occur. 1: Arbitration loss occurred.
4	IEBBnALTF	Broadcast communication flag 0: Individual communication status 1: Broadcast communication status
3	IEBBnACKF	Acknowledge transmission flag 0: A NACK signal is transmitted. 1: An $\overline{\text{ACK}}$ signal is transmitted.
2	IEBBnLCKF	Lock status flag 0: Unlocked 1: Locked

(a) Slave request flag for the unit (IEBBnSRQF): Bit 6

- Set/clear condition

  Set:

  When the unit is requested as a slave (if the condition in *Table 30-27 "Slave request conditions (conditions for setting the IEBBnSRQF bit)"* is satisfied), this flag is set (to 1) by hardware when the parity bit communication of the slave address field ends[a].

  Clear:

  This flag is cleared (to 0) by hardware when the unit is not requested as a slave (if the condition in *Table 30-27 "Slave request conditions (conditions for setting the IEBBnSRQF bit)"* is not satisfied). The timing[a] is the same as that for setting the flag.

a) The bit is updated when the communication of the slave address field parity bit finishes without an error such as a parity error occurring. For example, if the slave address reception parity is incorrect, the IEBBnSRQF bit is not updated and the previous value is retained.

Table 30-27 Slave request conditions (conditions for setting the IEBBnSRQF bit)

Status of unit	Received master address	Communication mode	Received slave address
Not locked	don't care	Individual	IEBBnUAR match
		Broadcast	Group matching
			FFFH match
Locked	Locked master matching	Individual	IEBBnUAR match
		Broadcast	Group matching
			FFFH match

Note IEBBnUAR match: When the reception slave address and unit IEBBnUAR register match

Group match: When the reception-slave-address group address and unit IEBBnUAR-register group address match

FFFH match: When the reception slave address is FFFH

Table 30-28 $\overline{\text{ACK}}$ signal response condition for the slave address field

Status of unit	Received master address	Communication mode	Received slave address
don't care	don't care	Individual	IEBBnUAR match

Note IEBBnUAR match: When the reception slave address and unit IEBBnUAR register match

Caution If a unit other than the locked master communicates with the unit while the unit is locked, the IEBBnSRQF bit is not set but the $\overline{\text{ACK}}$ signal is returned to the slave address field. This is because communication must be continued, even for communication of a unit other than the locked master, if the control data received using the control field is a slave status transmission request.

(b) Arbitration result flag (IEBBnARBF): Bit 5

- Set/clear condition

  Set:
  When the data output by the unit does not match the received data during the arbitration period[a]
  Clear: After each communication frame start bit is transmitted or received

a) In the FIFO mode, the arbitration loss flag is set (to 1) based on the same condition even while the master request flag (IEBBnBCR.IEBBnMSRQ) is retained.

Note that, if arbitration is lost in the FIFO mode, a NACK reception error is detected by returning the NACK signal after control data reception, and software processing by outputting IEBBTERR is required.

In the case of a status transmission request, the NACK signal is not returned because there is an automatic response.

The IEBBnARBF bit is set (to 1) if there is inter-unit-data contention during the arbitration period (the broadcast field and master address field period) and the unit loses arbitration.

Arbitration loss is judged to have occurred when the unit output data does not match the received data.

Because the IEBus controller uses AND logic, the unit that outputs 0 wins arbitration.

In other words, among units that output broadcast data (0) for the broadcast field, the unit that has the smallest master address wins arbitration.

Caution In the single mode, when the start interrupt occurs upon issuing a master request, use the IEBBnARBF bit to check for arbitration loss. To transfer data again in the case of arbitration loss, perform software processing to issue another master request.

In the FIFO mode, another master request is issued even if arbitration is lost the number of times specified by the IEBBnTMS.IEBBnALC0 to IEBBnALC2 bits. If the number of arbitration losses exceeds this setting, an interrupt occurs to indicate an arbitration loss error. To transfer data again, perform software processing to issue another master request.

(c) Broadcast communication flag (IEBBnALTF): Bit 4

Flag indicating whether the unit is performing broadcast communication. The contents of the flag are updated in the broadcast field of each frame.

- Set/clear condition

  Set: When “broadcast” is received by the broadcast field
  Clear: When individual is received by the broadcast field

Caution The broadcast flag is updated regardless of whether IEBus is the communication target.

In the FIFO mode, storage is performed upon header completion if reading the data received during the previous communication has finished (if the IEBBnBSR.IEBBnRFLF bit = 0 and the nBSR.IEBBnSRFP4 to IEBBnSRFP0 bits = 00000). Until reading the received data finishes, the IEBBnALTF register is not updated.

(d) Acknowledge transmission flag (IEBBnACKF): Bit 3

This flag indicates whether the $\overline{\text{ACK}}$ signal was transmitted during the acknowledge bit period of the acknowledge bit field when IEBus is the receiving unit.

- Set/clear condition

  Set:
  When $\overline{\text{ACK}}$ is transmitted upon the completion of the acknowledge bit period for each field
  Clear:
  When NACK is transmitted upon the completion of the acknowledge bit period for each field

Cautions

1. If a communication error occurs and the unit returns to the initial status, no update is performed at the end of the acknowledge bit period for the corresponding field. For example, if the reception parity for the control field is incorrect, because IEBBn changes to the initial status (the communication standby status) after parity reception due to the parity error, a NACK signal is returned by using the control field (more accurately, no $\overline{\text{ACK}}$ signal is returned), but this is not applied to the IEBBnACKF bit, and the previous value is retained.
2. In the single mode, because the occurrence timing (when IEBBTSTA or IEBBTV becomes active) for the start interrupt and status transmission interrupt is when parity bit reception ends, the reading of the IEBBnUSR register by the previously described interrupt in the slave mode during interrupt handler processing might overlap with the changing of the IEBBnACKF bit.

(e) Lock status flag (IEBBnLCKF): Bit 2

A flag that indicates whether the unit is locked.

- Set/clear condition

  Set:

  When an individual communication frame ends, lock-related data (3H, 6H, AH, and BH) is received by using the control field, the communication completion flag (the IEBBnISR.IEBBnETRF bit) is cleared (to 0), and the frame completion flag (the IEBBnISR.IEBBnEFMF bit) is s et (to 1).

  Clear:

  When an individual communication frame ends, lock-related data (3H, 6H, AH, and BH) is received by using the control field, and the communication completion flag (the IEBBnISR.IEBBnETRF bit) is set (to 1)

Cautions

1. Locking and unlocking are performed only during individual communication, not during broadcast communication.
2. While the master is locked, communication from a unit other than the master is not generally acknowledged. However, as an exception, if the control data (0H, 4H, or 5H) of a slave status transmission request is received, the communication is acknowledged even if it is not from the locked master. Note that, at this time, only a status request interrupt occurs, not a start interrupt or completion interrupt.

(19) IEBBnISR - IEBBn interrupt status register

The IEBBnISR status register indicates the interrupt source when an IEBBTSTA, IEBBTERR, or IEBBTV interrupt occurs.

Each time the IEBBTSTA, IEBBTERR, and IEBBTV interrupts occur, the IEBBnISR register is read, and the specified interrupt servicing is performed.

Access Only bit 6 can be read or written in 8 or 1-bit units.

Bits other than 6 are read-only, in 8 or 1-bit units.

Address <IEBBn_base> + 0048H

Initial value 00H

The IEBBnIEBE bit is reset when 0 is written to the IEBBnBCR.IEBBnPW bit.

Bits other than IEBBnIEBE are reset when the value of the IEBBnPW bit is overwritten with a different value.

Caution Be sure to set bits 1, and 7 to 0.

7	6	5	4	3	2	1	0
0	IEBBn IEBE	IEBBn STRF	IEBBn STSF	IEBBn ETRF	IEBBn EFMF	0	IEBBn FOVE
R	R/W[a]	R	R	R	R	R	R

a) Only the IEBBnIEBE bit can be written. Note that, when writing to the IEBBnIEBE bit, the bit can only be cleared (to 0).1Even if 1 is written, the IEBBnIEBE bit is not set (to 1).

Table 30-29 IEBBnISR register contents

Bit position	Bit name	Function
6	IEBBnIEBE	Communication error flag 0: No communication error has occurred. 1: A communication error has occurred.
5	IEBBnSTRF	Start interrupt flag 0: No start interrupt has occurred. 1: A start interrupt has occurred.
4	IEBBnSTSF	Status transmission flag (slave) 0: There is no status transmission request. 1: There is a status transmission request.
3	IEBBnETRF	Communication completion flag 0: Communication of the number of transmission bytes specified by the message length field has not finished. 1: Communication of the number of transmission bytes specified by the message length field has finished.
2	IEBBnEFMF	Frame completion flag 0: The frame (communication of the maximum number of transmission bytes[a]) has not finished. 1: The frame (communication of the maximum number of transmission bytes[a]) has finished.
0	IEBBnFOVE	Frame over error flag 0: No frame over error has occurred. 1: A frame over error has occurred.

a) Mode 1: 32 bits, mode 2: 128 bits

(a) Communication error flag (IEBBnIEBE): Bit 6

A flag that indicates a communication error has occurred.

- Set/clear condition

  Set:

  – Single mode:

    When a timing error, parity error (except in the data field during individual reception), NACK reception error, underrun error, or overrun error (during broadcast reception) occurs

  – FIFO mode:

    When a timing error, parity error (except in the data field during individual reception), NACK reception error, underrun error, overrun error (during broadcast reception), or arbitration loss error occurs

  Clear:

  – Single mode or FIFO mode:

    By software (The flag is cleared (to 0) when 0 is written to the IEBBnIEBE bit.)

When a communication error occurs, IEBBTERR IEBBTV occur in the single mode, and IEBBTERR occurs in the FIFO mode.

It is possible to determine what caused the error by reading the IEBBnESR and IEBBnISR registers.

(b) Start interrupt flag: (IEBBnSTRF): Bit 5

A flag that indicates the start interrupt.

- Set/clear condition

  Set:

  – Single mode:

  The flag is set (to 1) during master unit operation, regardless of whether arbitration is won or lost.

  The flag is set (to 1) during slave unit operation if there is a slave request (when the IEBBnUSR.IEBBnSRQF bit = 1) from the master (only the locked master when the unit is locked).

  The flag is set upon the completion of the slave address field parity period in all cases.

  – FIFO mode: The IEBBnSTRF bit is not set (to 1).

  Clear:

  – Single mode:

  If the unit is the communication target (during communication with the master unit or slave unit), the flag is cleared (to 0) by hardware when a status transmission interrupt, communication completion interrupt, frame completion interrupt, transmission data write request interrupt, reception data read interrupt, or communication error interrupt occurs.

  – FIFO mode: The IEBBnSTRF bit is normally cleared.

In the single mode, IEBBTSTA and IEBBTV occur when a start interrupt occurs. In the FIFO mode, the start interrupt, IEBBTSTA, and IEBBTV do not occur.

Cautions

1. When a start interrupt occurs, read the IEBBnUSR register to check the slave request flag (IEBBnSRQF) and arbitration result flag (IEBBnARBF) for the unit.
2. If the arbitration result flag (IEBBnARBF) is set (to 1) when a start interrupt occurs after the unit issues a master request, perform software processing to reissue the master request.

(c) Status transmission flag (slave) (IEBBnSTSF): Bit 4

This flag indicates that the master requested transmission of the slave status and lock address (higher 4 bits and lower 8 bits) when the controller was serving as a slave.

- Set/clear condition

  Set:

  – Single mode:

  When the unit is not locked, there is a slave request from any master, and 0H or 6H is received by using the control field

  When the unit is locked, there is a slave request from the locked master, and 0H, 4H, 5H, or 6H is received by using the control field, or when 0H, 4H, or 5H is received from a unit other than the locked master by using the control field

  The flag is set upon the completion of the control field parity period in all cases.

  For details, see *Table 30-30 “Conditions for setting the status transmission request flag (slave)”*.

  – FIFO mode: The IEBBnSTSF bit is not set (to 1).

  Clear:

  – Single mode:

  If the unit is the communication target (during communication with the master unit or slave unit), the flag is cleared (to 0) by hardware when a start interrupt, communication completion interrupt, frame completion interrupt, transmission data write request interrupt, reception data read interrupt, or communication error interrupt occurs.

  – FIFO mode: The IEBBnSTSF bit is normally cleared.

In the single mode, IEBBTSTA and IEBBTV occur when there is a status transmission request. In the FIFO mode, no interrupt signal is generated even if there is a status transmission request.

Caution Even if the slave transmission enable flag (the IEBBnBCR.IEBBnSTXE bit) is set to the prohibited value (0), the IEBBnSTSF bit is set (to 1).

Table 30-30 Conditions for setting the status transmission request flag (slave)

Various statuses					Value received using the control field				
equa	lockf	eqpa	IEBBn STXE	IEBBn SRXE	0H	3H, 7H	4H, 5H	6H	AH, BH, EH, FH
1	0	0	Any	Any	Set	Not set	Not set	Set	Not set
1	0	1	Any	Any	Set	Not set	Not set	Set	Not set
1	1	0	Any	Any	Set	Not set	Set	Not set	Not set
1	1	1	Any	Any	Set	Not set	Set	Set	Not set

Note equa: Unit match (during individual communication, IEBBnUAR register match, during broadcast communication: group match, FFF match)

lockf: Whether there is a lock

eqpa: Lock master match

IEBBnSTXE: Slave transmission enable flag (IEBBnBCR register bit 4)

IEBBnSRXE: Slave reception enable flag (IEBBnBCR register bit 3)

If a slave status transmission request interrupt occurs in the single mode, read the IEBBnCDR register to check the received control data contents, and then write the required slave status information to the IEBBnDR register.

The received control data and data written to the IEBBnDR register are shown below.

Table 30-31 Received control data and data written to the IEBBnDR register

Received control data	Function	Data written to the IEBBnDR register
0H, 6H	Slave status transmission	Value read from the IEBBnSSR register
4H	Transmission of the lower 8 bits of the lock address	Lower 8 bits of the IEBBnPAR register
5H	Transmission of the higher 8 bits of the lock address	Higher 8 bits of the IEBBnPAR register

Caution After a slave status transmission request interrupt occurs, be sure to write the appropriate data to the IEBBnDR register before the completion of the message length field. If writing is not in time, do not transmit IEBBnDR register data and cause an underrun error.

(d) Communication completion flag (IEBBnETRF): Bit 3

A flag that indicates whether communication ends after the number of bytes set in the message length field have been transferred.

- Set/clear condition

  Set:

  – Single mode:

  When the unit is the communication target (during communication with the master unit or slave unit) and the value of the IEBBnSCR register becomes 0 at the end of the data field acknowledge period

  Clear:

  – Single mode:

  The flag is cleared (to 0) by hardware when a start interrupt, status transmission interrupt, frame completion interrupt (when a communication completion interrupt does not occur), transmission data write request interrupt, reception data read interrupt, or communication error interrupt occurs.

In the single mode, IEBBTSTA and IEBBTV occur when the communication completion flag is set (to 1).

In the FIFO mode, the IEBBTSTA interrupt occurs at the same timing as in the single mode.

In the FIFO mode, reading the IEBBnETRF bit is prohibited. If the bit is read, the returned value is undefined.

In the FIFO mode, after IEBBTSTA occurs, transmission and reception can be controlled by performing the software processing below.

- When reception ends

1. Detecting the completion of communication based on the occurrence of the IEBBTSTA interrupt
2. Checking whether communication or a frame has finished by using the IEBBnFSR.IEBBnRTRF bit
3. Checking the number of received data bytes by using the IEBBnBSR.IEBBnSRFP4 to IEBBnSRFP0 bits (the number of bytes that have not been read)
4. Reading the received data from the IEBBnDR register (the reception FIFO buffer)

- When transmission ends

1. Detecting the completion of communication based on the occurrence of the IEBBTSTA interrupt
2. Checking whether communication or a frame has finished by using the IEBBnFSR.IEBBnTTRF bit
3. Checking the number of transmission data bytes by using the IEBBnBSR.IEBBnSTFP4 to IEBBnSTFP0 bits (the number of bytes that have not been transmitted)
4. Proceeding to retransmission processing if there is data that has not been transmitted

(e) Frame completion flag (IEBBnEFMF): Bit 2

This flag indicates whether communication ends after the maximum number of bytes (mode 1: 32 bytes, mode 2: 128 bytes) have been transferred.

- Set/clear condition

  Set:

  – Single mode:

  When the unit is the communication target (during communication with the master unit or slave unit) and the value of the IEBBnCCR register becomes 0 at the end of the data field acknowledge period

  Clear:

  – Single mode:

  The flag is cleared (to 0) by hardware when a start interrupt, status transmission interrupt, communication completion interrupt (when a frame completion interrupt does not occur), transmission data write request interrupt, reception data read interrupt, or communication error interrupt occurs.

In the single mode, IEBBTSTA and IEBBTV occur when the frame completion flag is set (to 1).

Cautions

1. In the single mode, if the IEBBnSCR and IEBBnCCR registers are both cleared to 00H at the end of the data field acknowledge period, the IEBBnETRF and IEBBnEFMF bits are set (to 1) at the same time.
2. In the single mode, if the last data field is a NACK signal when the maximum number of bytes that can be transmitted is reached during retransmission, the IEBBnEFMF and IEBBnIEBE bits (NACK reception error) are set (to 1) at the same time.

In the FIFO mode, the IEBBTSTA interrupt occurs at the same timing as in the single mode.

In the FIFO mode, reading the IEBBnEFMF bit is prohibited. If the bit is read, the returned value is undefined.

After IEBBTSTA occurs, transmission and reception can be controlled by performing the software processing below.

- When reception ends

1. Detecting the completion of communication based on the occurrence of the IEBBTSTA interrupt
2. Checking whether communication or a frame has finished by using the IEBBnFSR.IEBBnRTRF bit
3. Checking the number of received data bytes by using the IEBBnBSR.IEBBnSRFP4 to IEBBnSRFP0 bits (the number of bytes that have not been read)
4. Reading the received data from the IEBBnDR register (the reception FIFO buffer)

- When transmission ends

1. Detecting the completion of communication based on the occurrence of the IEBBTSTA interrupt
2. Checking whether communication or a frame has finished by using the IEBBnFSR.IEBBnTTRF bit
3. Checking the number of transmission data bytes by using the IEBBnBSR.IEBBnSTFP4 to IEBBnSTFP0 bits (the number of bytes that have not been transmitted)
4. Proceeding to retransmission processing if there is data that has not been transmitted

(f) Frame over error flag (IEBBnFOVE): Bit 0

This flag indicates that a frame over error has occurred.

- Set/clear condition

  Set:

  – Single mode: The IEBBnFOVE bit is not set (to 1).

  – FIFO mode:

    If the next broadcast communication is received as a slave unit before reading the data received during the previous communication finishes, the flag is set (to 1) after the first data field parity period finishes.

  Clear:

  – Single mode or FIFO mode:

    By software

    (The flag is cleared (to 0) when 1 is written to the IEBBnSTC1.IEBBnCLFE bit.)

In the single mode, the IEBBnFOVE bit is not set (to 1), and no interrupt requests occur.

In the FIFO mode, IEBBTERR occurs when the IEBBnFOVE bit is set (to 1).

Even if a frame over error occurs for the current frame, the data received during the previous frame is valid.

(20) IEBBnESR - IEBBn error status register

The IEBBnESR register is used to indicate the cause when an IEBus controller communication error interrupt occurs. Each bit of the IEBBnESR register is set (to 1) as soon as the communication error flag of the IEBBnISR register (IEBBnIEBE) is set (to 1). The cause of a communication error, if any, can be identified by checking the contents of the IEBBnESR register. (If the IEBBnISR.IEBBnIEBE bit is already set to 1, only the bits of the IEBBnESR register are set (to 1).)

Note that the bits can only be set (to 1) by hardware and are cleared by writing 1 to the IEBBnSTC0 register.

When 1 is written to the IEBBnSTC0 register, if there is contention with the hardware trying to specify (1), the hardware is prioritized.

Note that writing to the IEBBnESR register is invalid.

Access This register is read-only, in 8-bit or 1-bit units.

Address <IEBBn_base> + 004CH

Initial value 00H

This register is reset when the value of the IEBBnBCR.IEBBnPW bit is overwritten with a different value.

Caution When a communication error occurs, IEBBn returns to the initial status and prepares for the next communication. Regardless of whether an error occurs, if the next communication is started without handling errors, any error flags that have been set remain set. (If the system returns to the initial status due to a timing error, and a parity error is received during the next communication, both the timing error and parity error bits of the IEBBnESR register are set to 1.)

Therefore, handle any errors that occur before the next communication starts.

7	6	5	4	3	2	1	0
IEBBn TIME	IEBBn PARE	IEBBn NACE	IEBBn UNRE	IEBBn OVRE	0	IEBBn ABTE	IEBBn TRDE
R	R	R	R	R	R	R	R

Table 30-32 IEBBnESR register contents (1/2)

Bit position	Bit name	Function
7	IEBBnTIME	Timing error occurrence flag 0: No timing error has occurred. 1: A timing error has occurred.
6	IEBBnPARE	Parity error occurrence flag 0: No parity error has occurred. 1: A parity error has occurred.
5	IEBBnNACE	NACK reception error flag 0: No NACK reception error has occurred. 1: A NACK reception error has occurred.
4	IEBBnUNRE	Underrun error occurrence flag 0: No underrun error has occurred. 1: An underrun error has occurred.

Table 30-32 IEBBnESR register contents (2/2)

Bit position	Bit name	Function
3	IEBBnOVRE	Overrun error occurrence flag 0: No overrun error has occurred. 1: An overrun error has occurred.
1	IEBBnABTE	Arbitration loss error occurrence flag 0: The number of arbitration losses specified for the arbitration loss count setting bits (IEBBnTMS.IEBBnALC2 to IEBBnALC0) have not occurred. 1: The number of arbitration losses specified for the arbitration loss count setting bits (IEBBnTMS.IEBBnALC2 to IEBBnALC0) have occurred.
0	IEBBnTRDE	Inter-third-party communication error occurrence flag 0: An error occurred during communication targeting the unit. 1: An error occurred during inter-third-party communication.

(a) Timing error occurrence flag (IEBBnTIME): Bit 7

- Set condition

  Set: This flag is set (to 1) if a timing error occurs.

A timing error occurs if the high-/low-level width of the communication bit is not the defined value.

The defined value of the high- and low-level width is set to the bit processing block and monitored by the internal timer.

(b) Parity error occurrence flag (IEBBnPARE): Bit 6

- Set condition

  Set: This flag is set (to 1) if a parity error occurs.

When the unit is the reception unit (including while communication between others is being monitored), a parity error occurs when the parity data generated by the data received using the master address field, slave address field, control data field, or message length field does not match the received parity data.

However, when there is a data field mismatch, a NACK signal is returned and a data retransmission request is issued during individual communication, but a parity error occurs during broadcast communication.

Note During the above parity period, if the parity data received on the transmission side is inverted for some reason, a timing error occurs and communication ends.

Table 30-33 Operation when the parity data does not match

Field	Communication mode	Operation when the parity data does not match
Master address field	Individual/broadcast	A parity error occurs.
Slave address field	Individual/broadcast	A parity error occurs.
Control data field	Individual/broadcast	A parity error occurs.
Message length field	Individual/broadcast	A parity error occurs.
Data field	Individual	A NACK signal is returned to request retransmission.
	Broadcast	A parity error occurs.

(c) NACK reception error flag (IEBBnNACE): Bit 5

- Set condition

  Set: This flag is set (to 1) if a NACK reception error occurs.

A NACK reception error occurs if a NACK signal is received during the acknowledge bit period of the slave address field, control data field, or message length field during individual communication, regardless of whether the controller is operating as the master or slave.

When a NACK signal is received for the data field, no NACK reception error generally occurs because this reception signals a data retransmission request. However, if the last data field is a NACK signal, a NACK reception error does occur.

During reception, a NACK reception error is judged to have occurred if an output NACK signal is received for the last data of the slave address field, control data field, message length field, or data field.

Note that, during broadcast communication, no NACK reception errors occur because $\overline{\text{ACK}}$/NACK signal judgment is not performed.

No NACK reception errors occur during inter-third-party communication because only timing/parity errors are detected as errors. However, for the slave address field, NACK reception error judgment is performed because communication is participated in as a slave.

Table 30-34 NACK reception error judgment period

Communication mode		Slave address field	Control data field Data field	Message length field	Data field (last)
Individual communication	Master transmission	Occurs	Occurs	Occurs	Occurs
	Master reception	Occurs	Occurs	Occurs	Occurs
	Slave transmission	Occurs	Occurs	Occurs	Occurs
	Slave reception	Occurs	Occurs	Occurs	Occurs
	Inter-third-party communication	Occurs	Does not occur	Does not occur	Does not occur
Broadcast communication	All	Does not occur	Does not occur	Does not occur	Does not occur

(d) Underrun error occurrence flag (IEBBnUNRE): Bit 4

- Set condition

  Set:

  – Single mode:

  During data transmission using the data field, an underrun error occurs if writing the next data to be transmitted to the IEBBnDR register does not finish before the end of the data-field acknowledge bit period following the occurrence of IEBBTD, and this flag is set (to 1). However, if the NACK signal is received during the acknowledge bit period, no underrun error occurs because retransmission is performed.

  During inter-third-party communication, underrun errors do not occur because only timing/parity errors are detected as errors.

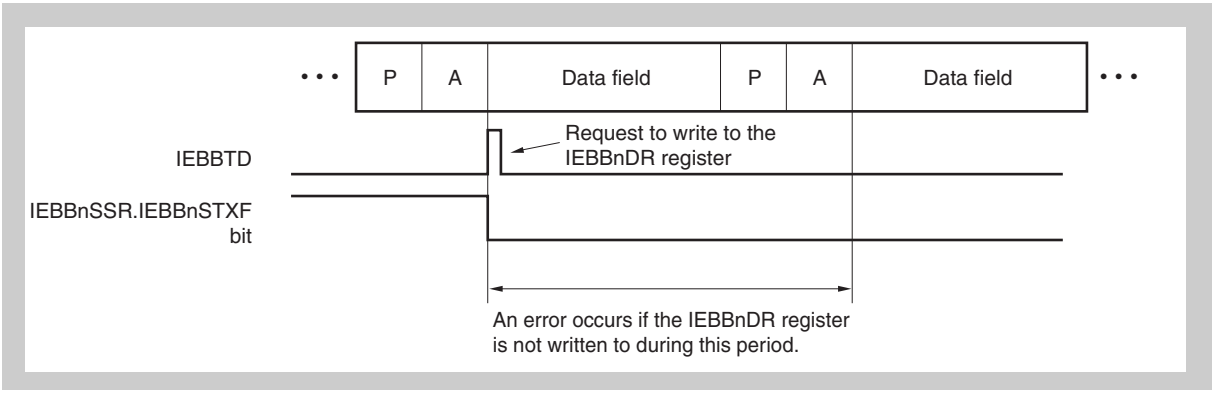


Figure 30-6 Underrun error occurrence timing

  – FIFO mode:

  Before data of the message length specified by the IEBBnTDL register is transmitted, if there are 0 data items to be transmitted remaining in the transmission FIFO buffer (if the IEBBnSSR.IEBBnSTXF bit = 0), an underrun error occurs if writing the next data to be transmitted to the IEBBnDR register does not finish before the end of the next data-field acknowledge bit period, and this flag is set (to 1). However, if the NACK signal is received during the acknowledge bit period, no underrun error occurs because retransmission is performed.

  During inter-third-party communication, underrun errors do not occur because only timing/parity errors are detected as errors.

(e) Overrun error occurrence flag (IEBBnOVRE): Bit 3

- Set condition

  Set:

  – Single mode:

  When the data field is used to receive data during broadcast communication, an overrun error occurs if reading the IEBBnDR register does not finish between when IEBBTD occurs and when the parity period of the data field finishes, and this flag is set (to 1).

  During individual communication, data retransmission is requested by returning a NACK signal without an error, and returning the NACK signal continues until the IEBBnDR register is read. (However, the frame ends when the maximum number of bytes that can be transferred is reached.)

  During inter-third-party communication, overrun errors do not occur because only timing/parity errors are detected as errors.

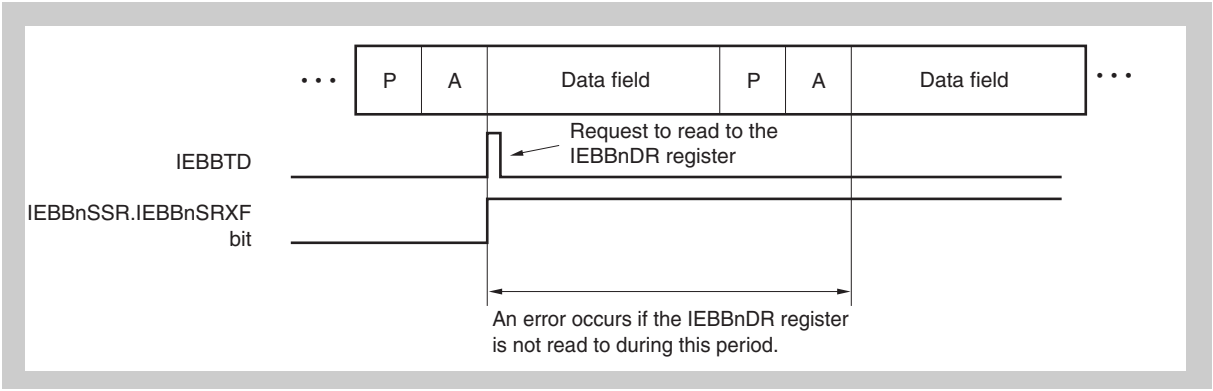


Figure 30-7 Overrun error occurrence timing

  – FIFO mode:

  During broadcast communication in mode 2, if there are 32 bytes of unread data in the reception FIFO buffer, an overrun error occurs if a 33rd byte of data is received, and this flag is set (to 1).

  During individual communication, data retransmission is requested by returning a NACK signal without an error, and returning the NACK signal continues until the IEBBnDR register is read. (However, the frame ends when the maximum number of bytes that can be transferred is reached.)

  If an overrun error occurs, do not read data by servicing interrupts.

  If an interrupt occurs while reading data, it might no longer be possible to read the correct data.

(f) Arbitration loss error occurrence flag (IEBBnABTE): Bit 1

- Set condition

  Set:

  If arbitration losses occur the number of times specified for the IEBBnTMS.IEBBnALC2 to IEBBnALC0 bits, this flag is set (to 1) when the last arbitration loss error occurs.

  For details about arbitration loss, see the description of the IEBBnARBF bit in *30.3.2 (18) "IEBBnUSR - IEBBn unit status register"*.

(g) Inter-third-party communication error occurrence flag (IEBBnTRDE): Bit 0

- Set condition

  Set:

  If an inter-third-party communication error occurs at the same time as a timing error or parity error that occurs during communication that is not related to the unit (inter-third-party communication), this flag is set (to 1) at the same time as the IEBBnTIME or IEBBnPARE bit.

Caution If an error occurs before the inter-third-party communication starts even when the slave address field does not match that of the unit (for example, if the NACK signal is received when the received address does not match that of the unit in the slave address field (if the IEBBnNACE bit is set (to 1))), the IEBBnTRDE bit is not set (to 1).

Note Communication between third parties may take place in the following two cases.

1. If the address received in the slave address field does not match that of the unit (during individual communication: matching with the IEBBnUAR register, during broadcast communication: matching with the group or FFFH) and communication continues after the $\overline{\text{ACK}}$ signal has been received, the unit monitors that communication.
2. If the unit cannot respond to the received control data in the control field during broadcast communication and if communication continues, the unit monitors that communication. For example, this happens when the unit receives the control data FH from the master during broadcast communication but the slave reception enable flag of the unit is disabled (IEBBnBCR.IEBBnSRXE bit = 0). (During individual communication, the NACK signal is returned and communication ends.)

(21) IEBBnFSR - IEBBn field status register

The IEBBnFSR register is used to store the field status state of the IEBus controller when various interrupts (IEBBTD, IEBBTSTA, IEBBTERR, and IEBBTV) occur.

Access This register is read-only, in 8-bit units.

Address <IEBBn_base> + 0050H

Initial value 00H

The IEBBnSSFS1 and IEBBnSSFS0 bits are reset when the IEBBnBCR.IEBBnPW bit is overwritten with a different value.

Cautions

1. If a different interrupt occurs before the IEBBnFSR register is read, the status information used at the time of the previous interrupt is overwritten with the status information used at the time of the new interrupt.
2. If an interrupt occurs during communication between third parties (during the reception of communication between other units), the IEBBnSSFS1 and IEBBnSSFS0 bits are cleared to 00. However, because the only interrupts that occur during communication between third parties are interrupts caused by errors, an inter-third-party communication error can be judged to have occurred by reading the inter-third-party communication error occurrence flag (the IEBBnTRDE bit) of the IEBBnESR register.
3. Even if the field status signal (an internal signal) changes, the IEBBnSSFS1 and IEBBnSSFS0 bits retain their previous values until an interrupt occurs.

7	6	5	4	3	2	1	0
IEBBn RTRF	IEBBn TTRF	0	0	0	0	IEBBn SSFS1	IEBBn SSFS0
R	R	R	R	R	R	R	R

Table 30-35 IEBBnFSR register contents

Bit position	Bit name	Function
7	IEBBnRTRF	Reception communication completion flag 0: Communication did not finish during reception. 1: Communication finished during reception.
6	IEBBnTTRF	Transmission communication completion flag 0: Communication did not finish during transmission. 1: Communication finished during transmission.
1, 0	IEBBnSSFS[1, 0]	For details about the IEBBnSSFS1 and IEBBnSSFS0 bits, see *Table 30-36 "Field status"*.

Table 30-36 Field status

Field status	Description		
	Master/slave	Field	Transmission/ reception
Slave reception status IEBBnSSFS1 and IEBBnSSFS0 bits = 00 (IEBBnFSR register = 00H)	Slave operation	Start bit	Reception
		Master address field	
		Slave address field	
		Control data field	
		Message length field	
		Data field	
Slave transmission status IEBBnSSFS1 and IEBBnSSFS0 bits = 01 (IEBBnFSR register = 01H)	Slave operation	Message length field	Transmission
		Data field	
Master reception status IEBBnSSFS1 and IEBBnSSFS0 bits = 10 (IEBBnFSR register = 02H)	Master operation	Message length field	Reception
		Data field	
Master transmission status IEBBnSSFS1 and IEBBnSSFS0 bits = 11 (IEBBnFSR register = 03H)	Master operation	Start bit	Transmission
		Master address field	
		Slave address field	
		Control data field	
		Message length field	
		Data field	

(a) Reception communication completion flag (IEBBnRTRF): Bit 7

This flag indicates that communication equivalent to the number of bytes specified by the message length has finished during reception.

- Set/clear condition

  Set:

  – Single mode: The IEBBnRTRF bit is not set (to 1).

  – FIFO mode:

  During reception, the flag is set (to 1) when the IEBBnISR.IEBBnETRF bit is set (to 1).

  Clear:

  – Single mode: The IEBBnRTRF bit always has the clear status.

  – FIFO mode:

  During reception, the flag is cleared (to 0) when the IEBBnISR.IEBBnEFMF bit is set (to 1).

Cautions

1. In the FIFO mode, if the set and clear conditions are both satisfied, setting the flag is prioritized.
2. The IEBBnRTRF bit is not cleared (to 0) by writing 1 to the IEBBnPCR.IEBBnCRPT bit.

(b) Transmission communication completion flag (IEBBnTTRF): Bit 6

This flag indicates that communication equivalent to the number of bytes specified by the message length has finished during transmission.

- Set/clear condition

  Set:

  – Single mode: The IEBBnTTRF bit is not set (to 1).

  – FIFO mode:

  During transmission, the flag is set (to 1) when the IEBBnISR.IEBBnETRF bit is set (to 1).

  Clear:

  – Single mode: The IEBBnTTRF bit always has the clear status.

  – FIFO mode:

  During transmission, the flag is cleared (to 0) when the IEBBnISR.IEBBnEFMF bit is set (to 1).

Cautions

1. In the FIFO mode, if the set and clear conditions are both satisfied, setting the flag is prioritized.
2. The IEBBnTTRF bit is not cleared (to 0) by writing 1 to the IEBBnPCR.IEBBnCTPT bit.

(c) Field status flags (IEBBnSSFS1 and IEBBnSSFS0): Bits 1 and 0

These flags store the state of the IEBus controller field status when various interrupts (IEBBTD, IEBBTSTA, IEBBTERR, and IEBBTV) occur.

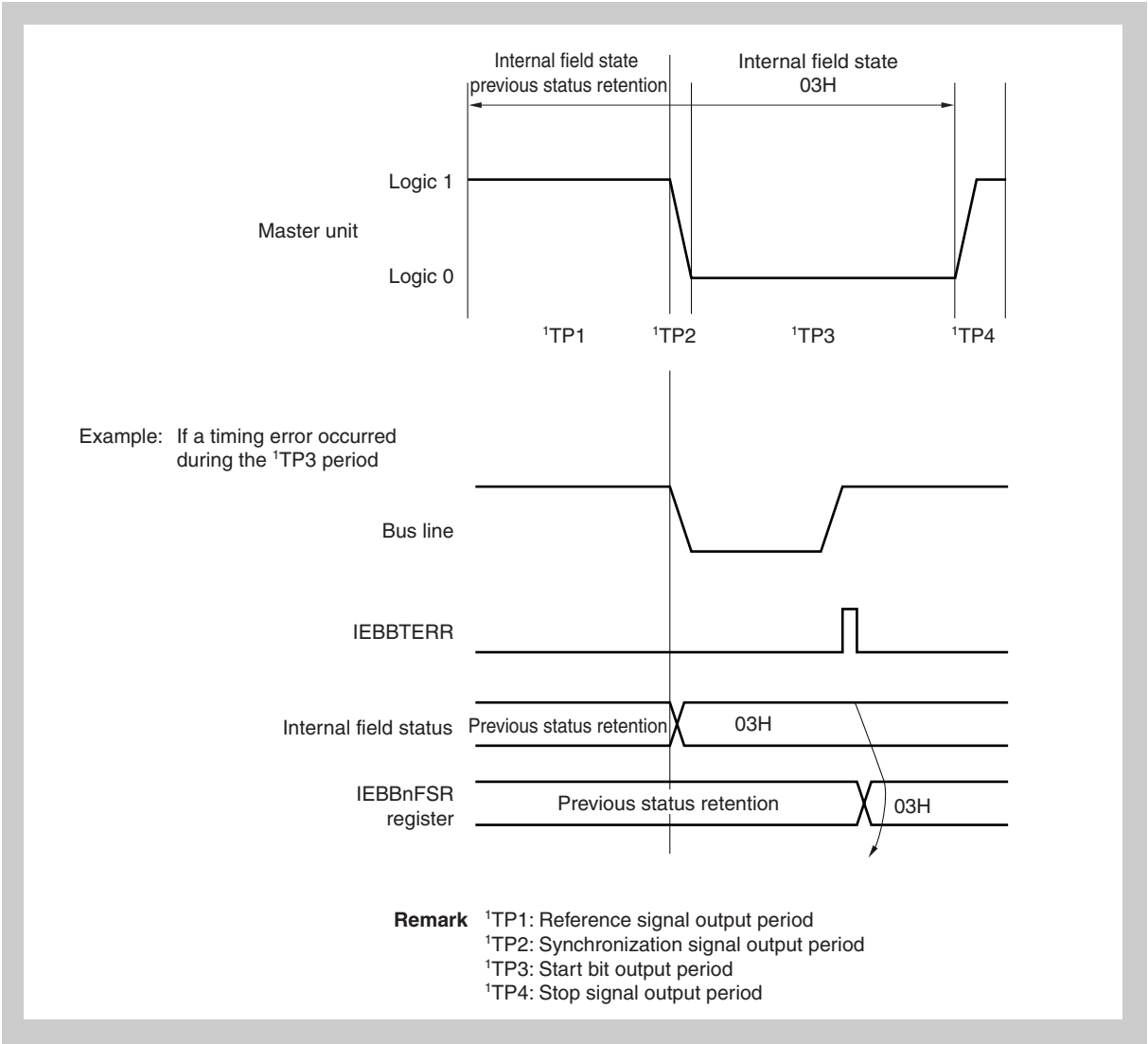


Figure 30-8 Start bit field status for the master (internal signal)

When the start bit shown in *Figure 30-8 "Start bit field status for the master (internal signal)"* is output for the master, the previous field status value is retained until ^{1}TP1. At point ^{1}TP2 and after, the field status value is 03H. If a timing error occurs at point ^{1}TP3 and IEBBTERR is output, 03H is stored in the IEBBnFSR register.

Because IEBBTERR does not occur if communication is performed normally, the field status value is not stored in the IEBBnFSR register, and the IEBBnFSR register retains the previous value at and after point ^{1}TP2.

(22) IEBBnSCR - IEBBn success count register

The IEBBnSCR register indicates the number of remaining communication bytes.

The value specified by the IEBBnDLR register is stored in the IEBBnSCR register after the message length field processing finishes, and the count value of the counter to be decremented according to the data field $\overline{ACK}$ signal is read.

In other words, because the number of successfully communicated bytes is subtracted from the number of data bytes to be communicated, the IEBBnSCR register indicates the remaining number of bytes to be communicated.

Note that the communication completion flag (the IEBBnISR.IEBBnETRF bit) is set (to 1) when the count value reaches 00H.

The data in the IEBBnSCR register is updated when the $\overline{ACK}$ signal is received at the end of the message length field parity period or data field acknowledge bit period.

Access This register is read-only, in 8-bit units.

Address <IEBBn_base> + 0054H

Initial value 01H

This register is reset when the value of the IEBBnBCR.IEBBnPW bit is overwritten with a different value.

Caution When 00H is read from the IEBBnSCR register, it is not possible to judge whether the remaining number of communication data bytes is 0 (indicating communication completion) or 256. Therefore, the communication completion flag (the IEBBnISR.IEBBnETRF bit) can be used with this register to make this judgment.

7	6	5	4	3	2	1	0
R	R	R	R	R	R	R	R

Table 30-37 IEBBnSCR register contents

Bit								Setting	Number of remaining communication data bytes
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	01H	1 byte
0	0	0	0	0	0	1	0	02H	2 bytes
...	...	...	...	...	...	...	...	...	...
0	0	0	1	0	1	0	0	20H	32 bytes
...	...	...	...	...	...	...	...	...	...
1	1	1	1	1	1	1	1	FFH	255 bytes
0	0	0	0	0	0	0	0	00H	0 bytes (communication completion) or 256 bytes

(23) IEBBnCCR - IEBBn communication count register

The IEBBnCCR register indicates the number of bytes remaining from the communication byte number specified by the communication mode.

This register indicates the number of transfer bytes.

The maximum number of transmitted bytes per frame defined in each mode (mode 1: 32 bytes, mode 2: 128 bytes) is preset to this register. The count value of the counter that is decremented during the acknowledge bit period of the data field regardless of the $\overline{\text{ACK}}$/NACK signal is read from this register. In contrast with the IEBBnSCR register, which is decremented when there is normal communication (the $\overline{\text{ACK}}$ signal), the IEBBnCCR register is decremented when one byte is communicated, regardless of the $\overline{\text{ACK}}$/NACK signal. Note that the frame completion flag (the IEBBnISR.IEBBnEFMF bit) is set (to 1) when the counter reaches 00H.

The preset value of the maximum number of transmitted bytes per frame is 20H (32 bytes) in mode 1 and 80H (128 bytes) in mode 2.

Access This register is read-only, in 8-bit units.

Address <IEBBn_base> + 0058H

Initial value 20H

This register is reset when the value of the IEBBnBCR.IEBBnPW bit is overwritten with a different value.

Caution The value of the IEBBnCCR register is not updated by writing to the IEBBnPSR.IEBBnCMD bit.

7	6	5	4	3	2	1	0
R	R	R	R	R	R	R	R

(24) IEBBnSTC0 - IEBBn status clear register 0

The IEBBnSTC0 register is used to clear the IEBBnESR register.

Access This register is write-only, in 1-bit units.

Address <IEBBn_base> + 005CH

Initial value 00H

7	6	5	4	3	2	1	0
IEBBn CLTM	IEBBn CLPA	IEBBn CLNC	IEBBn CLUR	IEBBn CLOV	0	IEBBn CLAB	IEBBn CLTR
W	W	W	W	W	W	W	W

Table 30-38 IEBBnSTC0 register contents

Bit position	Bit name	Function
7	IEBBnCLTM	This bit is used to clear the timing error flag (IEBBnESR.IEBBnTIME). 0: No operation 1: Clear the IEBBnTIME bit. Writing 1 is valid, and writing 0 does not change the internal status. When the bit is read, 0 is always returned.
6	IEBBnCLPA	This bit is used to clear the parity error flag (IEBBnESR.IEBBnPARE). 0: No operation 1: Clear the IEBBnPARE bit. Writing 1 is valid, and writing 0 does not change the internal status. When the bit is read, 0 is always returned.
5	IEBBnCLNC	This bit is used to clear the NACK reception error flag (IEBBnESR.IEBBnNACE). 0: No operation 1: Clear the IEBBnNACE bit. Writing 1 is valid, and writing 0 does not change the internal status. When the bit is read, 0 is always returned.
4	IEBBnCLUR	This bit is used to clear the underrun error flag (IEBBnESR.IEBBnUNRE). 0: No operation 1: Clear the IEBBnUNRE bit. Writing 1 is valid, and writing 0 does not change the internal status. When the bit is read, 0 is always returned.
3	IEBBnCLOV	This bit is used to clear the overrun error flag (IEBBnESR.IEBBnOVRE). 0: No operation 1: Clear the IEBBnOVRE bit. Writing 1 is valid, and writing 0 does not change the internal status. When the bit is read, 0 is always returned.
1	IEBBnCLAB	This bit is used to clear the arbitration loss error flag (IEBBnESR.IEBBnABTE). 0: No operation 1: Clear the IEBBnABTE bit. Writing 1 is valid, and writing 0 does not change the internal status. When the bit is read, 0 is always returned.
0	IEBBnCLTR	This bit is used to clear the inter-third-party communication error flag (IEBBnESR.IEBBnTRDE). 0: No operation 1: Clear the IEBBnTRDE bit. Writing 1 is valid, and writing 0 does not change the internal status. When the bit is read, 0 is always returned.

(25) IEBBnSTC1 - IEBBn status clear register 1

The IEBBnSTC1 register is used to clear the IEBBnISR.IEBBnFOVE bit.

Access This register is write-only, in 1-bit units.

Address <IEBBn_base> + 0060H

Initial value 00H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	IEBBn CLFF
W	W	W	W	W	W	W	W

Table 30-39 IEBBnSTC1 register contents

Bit position	Bit name	Function
0	IEBBnCLFF	This bit is used to clear the frame over error flag (IEBBnISR.IEBBnFOVE). 0: No operation 1: Clear the IEBBnFOVE bit. Writing 1 is valid, and writing 0 does not change the internal status. When the bit is read, 0 is always returned.

(26) IEBBnDR - IEBBn data register

The IEBBnDR register is used to set up the communication data. Specify the communication data (8 bits) for bits 7 to 0.

Notes

1. The IEBBnDR register consists of a write register and a read register. Therefore, data written to this register cannot be read as is. The data received during IEBus communication can be read.
2. In the FIFO mode, the data in the FIFO buffer can be transferred by continuously accessing the IEBBnDR register. (See *30.5.1 (1) "Transmission FIFO buffer"* for details about using the transmission unit or *30.5.1 (2) "Reception FIFO buffer"* for details about using the reception unit.)

(a) For the transmission unit

If the unit is the transmission unit (during master or slave transmission), the bits in the data field are transmitted as data bits starting with the highest bits when writing to the IEBBnDR register.

Specify the first byte of transmission data before starting communication (IEBBnBCR.IEBBnMSRQ bit = 0).

Even in the FIFO mode, write at least one byte of transmission data to the FIFO buffer before starting communication (IEBBnMSRQ bit = 0).

In the FIFO mode, the contents of the FIFO buffer are not reset in sync with the IEBBnBCR.IEBBnPW bit. Clearing the pointer value (to 0) eliminates the remaining data visible to the user. The stored data becomes undefined.

During transmission (master or slave transmission), if a data interrupt (IEBBTD) occurs, the next transmission data is written to the IEBBnDR register.

In the single mode, if a status transmission interrupt (IEBBTSTA or IEBBTV) occurs, the status data is written to the IEBBnDR register according to the control data.

(b) For the reception unit

The 1 byte of data received using the data field is read from the IEBBnDR register if the unit is the reception unit (master or slave reception). Storage is performed at the end of the data field parity period if the parity value is normal. The read value is reset by clearing the IEBBnPW bit to 0.

During reception (master or slave reception), if a data interrupt (IEBBTD) occurs, received data is read from the IEBBnDR register.

Access This register can be read or written in 8-bit units.

Address <IEBBn_base> + 0064H

Initial value 00H

The read value is reset when the IEBBnBCR.IEBBnPW bit is overwritten.

Cautions

1. If writing to the IEBBnDR register is not in time for transmission, an underrun error occurs and communication ends.
2. Write to the status data register IEBBnDR after a status transmission interrupt occurs and before the end of the message length field.
3. In the single mode, if the IEBBnDR register is not read before the next reception, the communication differs in the case of individual versus broadcast communication.
   - For individual communication, a NACK signal is returned for the field, and the master is requested to transmit the same data. Received data is not stored in the IEBBnDR register. If the NACK signal is returned again before the IEBBnDR register is read and the register has still not been read when the maximum number of transferable bytes is reached, frame completion (IEBBTSTA, IEBBTV) and a NACK reception error (IEBBTERR, IEBBTV) occur at the same time.
   - During broadcast communication, an overrun error occurs and communication ends. Received data is not stored in the IEBBnDR register. The overrun error flag (IEBBnOVRE) is set (to 1).
4. In the FIFO mode, if the next reception occurs before all the data received during the previous communication is read, the communication differs in the case of individual versus broadcast communication.
   - For individual communication, a NACK signal is returned for the data field, and the master is requested to transmit the same data. Received data is not stored in the FIFO buffer. If the NACK signal is returned again before all the data in the FIFO buffer is read and the FIFO buffer has still not been read when the maximum number of transferable bytes is reached, frame completion (IEBBTSTA) and a NACK reception error (IEBBTERR) occur at the same time.
   - During broadcast communication, an overrun error occurs and communication ends. Received data is not stored in the FIFO buffer. The frame over error flag (IEBBnISR.IEBBnFOVE) is set (to 1).
5. In the FIFO mode and mode 2, if there are 32 bytes of unread data and the next reception occurs, the communication differs in the case of individual versus broadcast communication.
   - For individual communication, a NACK signal is returned for the field, and the master is requested to transmit the same data. Received data is not stored in the FIFO buffer. If the NACK signal is returned again before the IEBBnDR register is read and reading the register has still not finished when the maximum number of transferable bytes is reached, frame completion (IEBBTSTA) and a NACK reception error (IEBBTERR) occur at the same time.
   - During broadcast communication, an overrun error occurs and communication ends. Received data is not stored in the FIFO buffer. The overrun error flag (IEBBnESR.IEBBnOVRE) is set (to 1).

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

30.4 Interrupt Operations

30.4.1 Interrupt request signals

Various interrupts occur in response to the eight interrupt requests below. The high level width of each interrupt signal is one PCLK clock cycle.

The interrupts that occur differ depending on whether the system is in the single or FIFO mode.

(1) Single mode

The causes of interrupts in the single mode are shown blow.

Table 30-40 Causes of interrupts in the single mode

Symbol	IEBBTD	IEBBTV	IEBBTERR	IEBBTSTA	Interrupt cause
IEBBnIEBE		Occurs	Occurs		Communication error (When the IEBBnISR.IEBBnIEBE bit = 1) Note that IEBBnIEBE occurs when the following bits of the IEBBnESR register = 1. • Timing error (IEBBnTIME) • Parity error (IEBBnPARE) • NACK reception error (IEBBnNACE)[a] • Underrun error (IEBBnUNRE) • Overrun error (IEBBnOVRE)
IEBBnSTRF		Occurs		Occurs	Start request (when the IEBBnISR.IEBBnSTRF bit = 1)
IEBBnSTSF		Occurs		Occurs	Status transmission request (when the IEBBnISR.IEBBnSTSF bit = 1)
IEBBnETRF		Occurs		Occurs	End of communication (When the IEBBnISR.IEBBnETRF bit = 1)
IEBBnEFMF		Occurs		Occurs	End of frame (When the IEBBnISR.IEBBnEFMF bit = 1)[a]
IEBBnFOVE					Frame over (When the IEBBnISR.IEBBnFOVE bit = 1)
WRREQ	Occurs				Transmission data write request (when the IEBBnSSR.IEBBnSTXF bit = 0)[b]
RDREQ	Occurs				Reception data read request (when the IEBBnSSR.IEBBnSRXF bit = 1) [c]

a) If the frame data ends with a NACK signal, IEBBTV and IEBBTSTA are triggered by setting the frame completion indicating bit IEBBnISR.IEBBnEFMF (to 1).
At this time, the IEBBTERR and IEBBTV interrupts are triggered by a NACK reception error.
Three interrupts (IEBBTV, IEBBTSTA, and IEBBTERR) occur at the same time.

b) During master transmission:

1. IEBBTD occurs after receiving the $\overline{\text{ACK}}$ signal, which follows message length field transmission. However, if the transfer size is one byte (the IEBBnTDL register = 01H), IEBBTD does not occur.

2. IEBBTD occurs after receiving the $\overline{\text{ACK}}$ signal, which follows data field transmission. However, IEBBTDdoes not occur before transmitting the final data, or after transmitting the final data and then receiving the ACK signal. More specifically, if the message length is five bytes, IEBBTD does not occur after transmitting the 4th or 5th byte. In addition, when transmitting the maximum number of transferable bytes for one frame, IEBBTD does not occur after transmitting byte number (maximum number of transferrable bytes – 1) or byte number (maximum number of transferable bytes).

During slave transmission:

1. IEBBTD occurs after receiving the $\overline{\text{ACK}}$ signal, which follows message length field transmission. However, if the transfer size is one byte or the received control bit is a status request (0H, 4H, 5H, or 6H), IEBBTD does not occur (and a status transmission interrupt occurs instead).

2. After data field transmission, the operation is the same as for *2* under During master transmission.

c) RDREQ occurs after receiving the parity bit by using the data field. However, if the self-transmitted parity bit differs from the received parity bit, there is a timing error and no interrupt occurs.

(2) FIFO mode

The causes of interrupts in the FIFO mode are shown below.

Table 30-41 Causes of interrupts in the FIFO mode

Symbol	IEBBTD	IEBBTV	IEBBTERR	IEBBTSTA	Interrupt cause
IEBBnIEBE			Occurs		Communication error (When the IEBBnISR.IEBBnIEBE bit = 1) Note that IEBBnIEBE occurs when the following bits of the IEBBnESR register = 1. • Timing error (IEBBnTIME) • Parity error (IEBBnPARE) • NACK reception error (IEBBnNACE)[a] • Underrun error (IEBBnUNRE) • Overrun error (IEBBnOVRE)[b] • Arbitration loss error (IEBBnABTE)
IEBBnSTRF					Start request (when the IEBBnISR.IEBBnSTRF bit = 1)
IEBBnSTSF					Status transmission request (when the IEBBnISR.IEBBnSTSF bit = 1)
IEBBnETRF		Occurs[c]		Occurs	End of communication (When the IEBBnISR.IEBBnETRF bit = 1)
IEBBnEFMF		Occurs[c]		Occurs	End of frame (When the IEBBnISR.IEBBnEFMF bit = 1)[a]
IEBBnFOVE			Occurs		Frame over (When the IEBBnISR.IEBBnFOVE bit = 1)[b]
WRREQ	Occurs[d]				Transmit data write request
RDREQ		Occurs[e]			Receive data write request Parity

a) If the frame data ends with a NACK signal, IEBBTV and IEBBTSTA are triggered by setting the frame completion indicating bit IEBBnISR.IEBBnEFMF (to 1).
Note that the IEBBTERR interrupt is triggered by a NACK reception error.
Three interrupts (IEBBTV, IEBBTSTA, and IEBBTERR) occur at the same time, regardless of transmission or reception.

b) If data is received during the broadcast communication for the next frame while the FIFO buffer is full due to the reception of the previous frame and the data has not been read, the IEBBnISR.IEBBnFOVE and IEBBnESR.IEBBnOVRE bits are set (to 1) at the same time.

c) IEBBTV occurs based on the same conditions and at the same timing as IEBBTSTA. (For details, see *Table 30-42 "ACK/NACK for IEBBnETRF and IEBBnEFMF"*.
For individual communication:
1. Transmission-side device

IEBBnETRF:
This is set (to 1) after receiving the ACK signal. If the NACK signal is received, communication does not end.

IEBBnEFMF: This is set (to 1) after reception regardless of the ACK/NACK signal.

2. Reception-side device

IEBBnETRF:
This is set (to 1) after transmitting the ACK signal. If the NACK signal is transmitted, communication does not end.

IEBBnEFMF:
This is set (to 1) after transmission regardless of the ACK/NACK signal.
The NACK signal is output when there is no room in the reception FIFO buffer or when there is a data retransmission request due to a parity mismatch. Not returning an ACK signal to the reception side as the bus status after exiting communications due to the detection of an error does not constitute a NACK signal. In this case, IEBBTSTA and IEBBTV do not occur because the system does not enter the frame completion status.

For broadcast communication:
For broadcast communication, no ACK signal is returned from the slave. Therefore, it is judged that the NACK signal was successfully returned regardless of master transmission or slave reception, and the IEBBnETRF or IEBBnEFMF interrupt occurs.

d) This occurs when the conditions specified by the IEBBnTMS.IEBBnSLTI1 and IEBBnSLTI0 bits are satisfied. The occurrence timing is after receiving the ACK signal, which follows data field transmission. However, if the transfer size is one byte or the received control bit is a status request (0H, 4H, 5H, or 6H), IEBBTD does not occur.

e) 1. Transmission-side device
RDREQ is not set under this condition.
2. Reception-side device
RDREQ is set (to 1) when the conditions specified by the IEBBnTMS.IEBBnSLRI1 and IEBBnSLRI0 bits are satisfied (after confirming that a normal parity bit value has been received).
If there is a parity bit mismatch, RDREQ is not set (to 1) because the conditions are not satisfied.
On normal completion or frame completion, RDREQ interrupt sources are masked.

Table 30-42 ACK/NACK for IEBBnETRF and IEBBnEFMF

Field status	Individual communication				Broadcast communication			
	IEBBnETRF		IEBBnEFMF		IEBBnETRF		IEBBnEFMF	
	ACK	NACK	ACK	NACK	ACK	NACK	ACK	NACK
Master transmission	Occurs	Does not occur	Occurs	Occurs	–	Occurs	–	Occurs
Master reception	Occurs	Does not occur	Occurs	Occurs	–	–	–	–
Slave transmission	Occurs	Does not occur	Occurs	Occurs	–	–	–	–
Slave reception	Occurs	Does not occur	Occurs	Occurs	–	Occurs	–	Occurs

Note The IEBBTV usage method is shown below.

1. Generating IEBBTV before communication finishes (when the IEBBnTMS.IEBBnSLRI1 and IEBBnSLRI0 bit settings are 32 bits or less)
   This operation is not generally performed in mode 1. (It can be performed but is not.) The operation is performed in mode 2.
   – Use IEBBTV to check the number of data bytes received by the FIFO buffer, and then perform a readout operation. (Be sure to check the number because the IEBBTV interrupt servicing is assumed to be late.)
   – For IEBBTSTA (communication completion/frame completion), check the status.
   – Because the status is changed after receiving data, the interrupt priority is IEBBTV followed by IEBBTSTA.
2. If not generating IEBBTV during communication (if the IEBBnSLRI1 and IEBBnSLRI0 bit settings are 32 bytes)

   Use mode 1. (Mode 2 can also be used.)
   – Mode 1

   Mask IEBBTV (so it is not used).
   Use IEBBTSTA (communication completion/frame completion) to check the number of data bytes in the FIFO buffer, and then check the readout and status changes.
   – Mode 2

   Perform the same operations as in *1*.

30.4.2 Interrupt judgment examples

Interrupt judgment examples for the single mode are shown below.

(1) When using IEBBTD

IEBBn transmission/reception must be checked by issuing an IEBBTD interrupt.

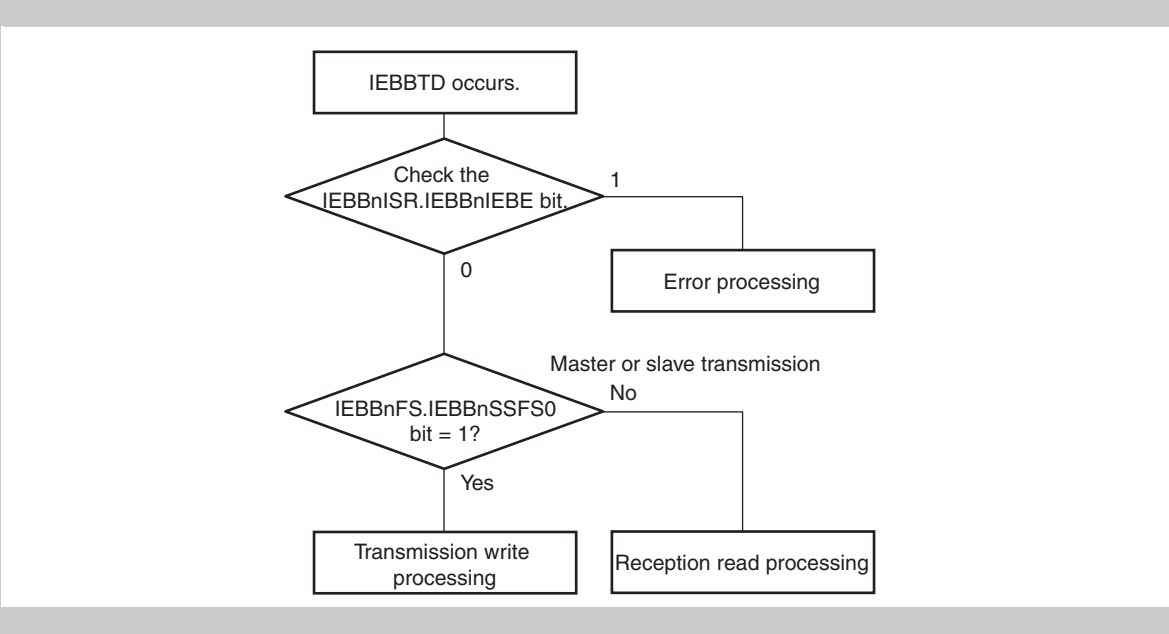


Figure 30-9 IEBBTD interrupt judgment example

Caution Even if IEBBTD occurs, an error might occur depending on when the interrupt is handled.

Such errors include timing errors after IEBBTD occurs. To increase data processing reliability, only handle data after using the IEBBnISR.IEBBnIEBE bit to make sure that no error has occurred.

(2) When using IEBBTERR

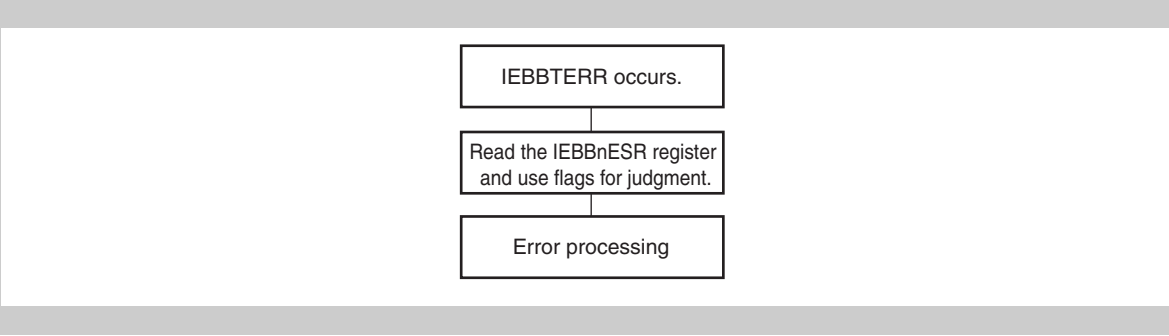


Figure 30-10 IEBBTERR interrupt judgment example

(3) When using IEBBTSTA or IEBBTV

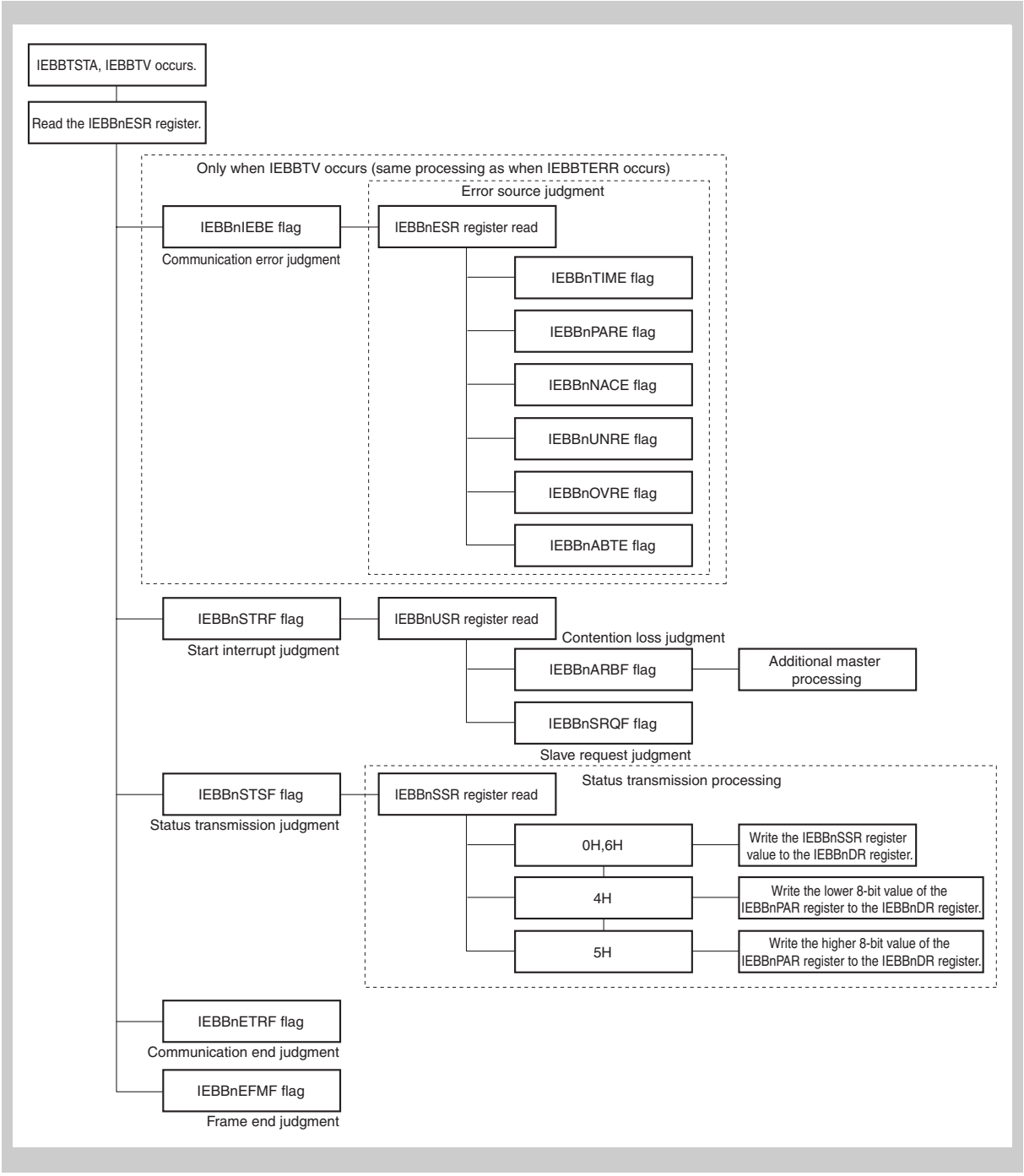


Figure 30-11 IEBBTSTA and IEBBTV interrupt judgment examples

30.5 Operation

30.5.1 FIFO

(1) Transmission FIFO buffer

When the IEBBnTMS.IEBBnFMDE bit = 1, data can be stored in the FIFO buffer while automatically incrementing the FIFO buffer pointer for writing by continuously writing transmission data to the IEBBnDR register. The FIFO buffer size is 8 bits x 32 stages.

When the transfer is started, the data indicated by the load pointer is transferred. After the transfer finishes, the load pointer is incremented. The initial value for the write pointer and load pointer is 0.

The IEBBnBSR.IEBBnTFLF bit is set (to 1) when there are 32 bytes of data in the FIFO buffer, and the transmission FIFO buffer overwrite flag (IEBBnBSR.IEBBnFOVW) is set (to 1) when a 33rd byte of data is written while the IEBBnTFLF bit = 1. At this time, the write data is ignored and the write pointer is not changed.

The data below can be written when one byte is transferred while the IEBBnTFLF bit = 1 and then the bit is cleared to 0.

If the write is not in time for data loading, an underrun error occurs.

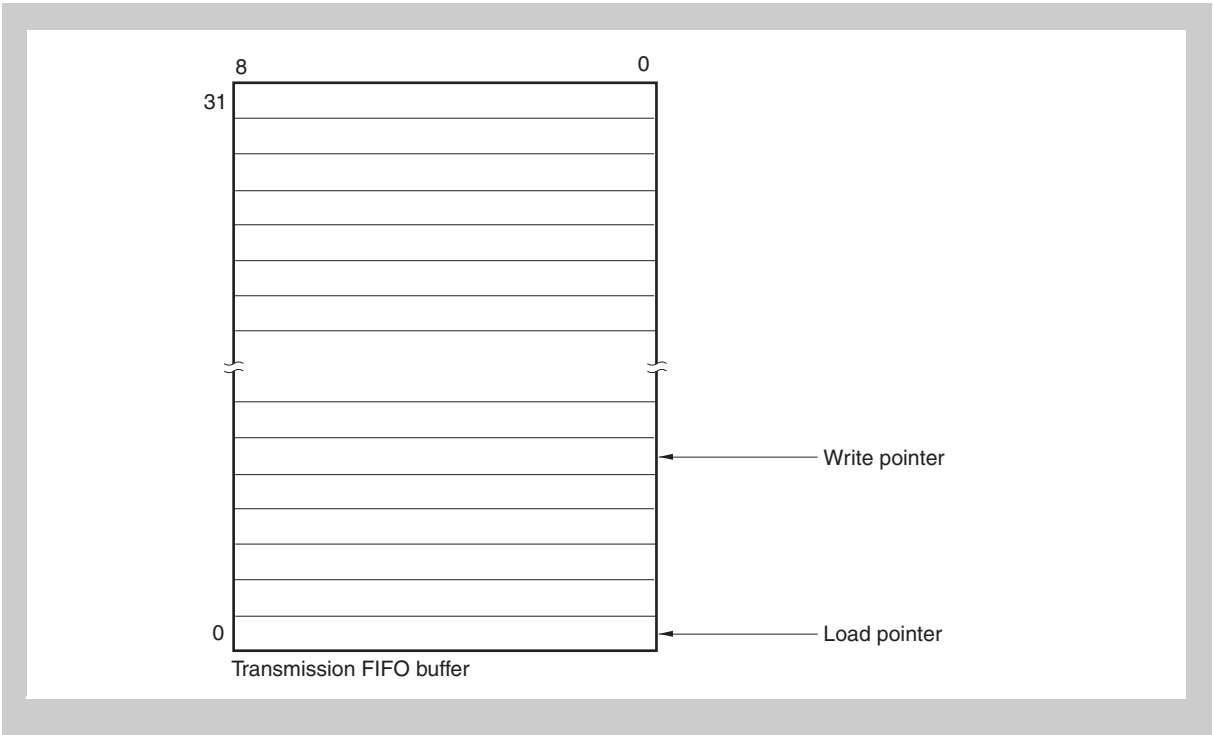


Figure 30-12 Transmission FIFO buffer

(2) Reception FIFO buffer

When the IEBBnTMS.IEBBnFMDE bit = 1, the received data is stored at the address indicated by the storage pointer. The storage pointer is incremented after storing the data. Data is stored in the FIFO buffer at the end of the data field parity period if the parity value is normal. The FIFO buffer size is 8 bits x 32 stages.

By reading the IEBBnDR register, the data in the FIFO buffer can be read while automatically incrementing the read pointer. The initial value for the read pointer and storage pointer is 0.

If there are 32 bytes of unread data in the FIFO buffer, the operation when the next data is received is as follows.

- During individual communication: No data is stored, a NACK signal is returned, and data retransmission is requested.
- During broadcast communication: No data is stored and an overrun error occurs.

  If the IEBBnDR register is read again after reading the received data that has been stored finishes (when the read pointer = the storage pointer), the read pointer is not changed, and the over-read flag (IEBBnBSR.IEBBnFOVR bit) is set (to 1).

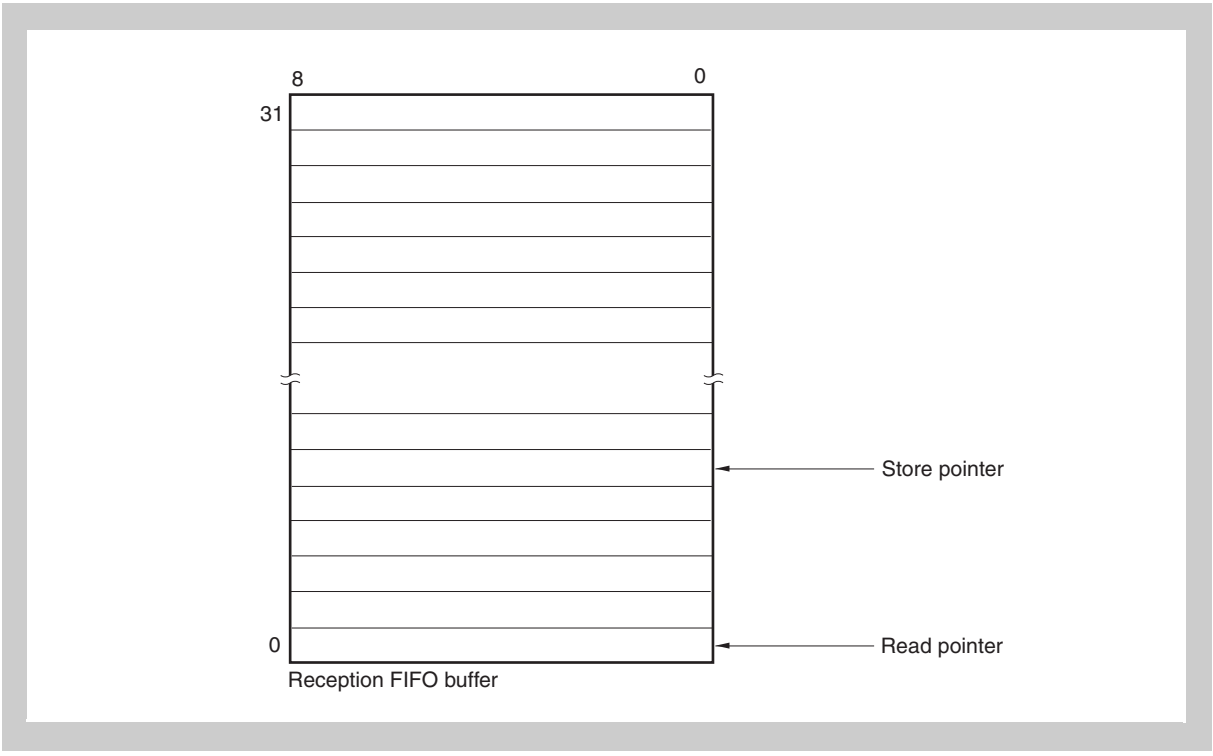


Figure 30-13 Reception FIFO buffer

30.5.2 Initial settings

After setting the IEBBnBCR.IEBBnPW bit to 1, set up the registers below, and then start communication processing.

Table 30-43 Initial setup

Register name	Function	Example
IEBBnPSR	Operation clock and communication mode settings	80H
IEBBnUAR	Set a unit address.	101H
IEBBnCKS	Clock Selection	15H
IEBBnTMS	Communication control	01H[a]

a) For use in the single mode, the initial values do no have to be changed.

30.5.3 Master transmission (single mode)

The unit transmits data and commands to the slave unit as the master.

Data interrupts are used to write transmission data to the IEBBnDR register for each one-byte transfer.

(1) Register settings

After specifying the initial settings in *30.5.2 "Initial settings"*, set up the registers below before starting communication.

Table 30-44 Standard initial processing

Register name	Function	Example
IEBBnSAR	Communication partner unit address	102H
IEBBnCDR or IEBBnTCD	Control data (AH, BH, EH, FH)	FH
IEBBnDLR	Message length	02H
IEBBnDR	Data (1st byte of data)	11H

Table 30-45 Communication startup processing

Register name	Function	Example
IEBBnBCR	Communication startup processing	C8H

(2) Interrupt occurrence timing

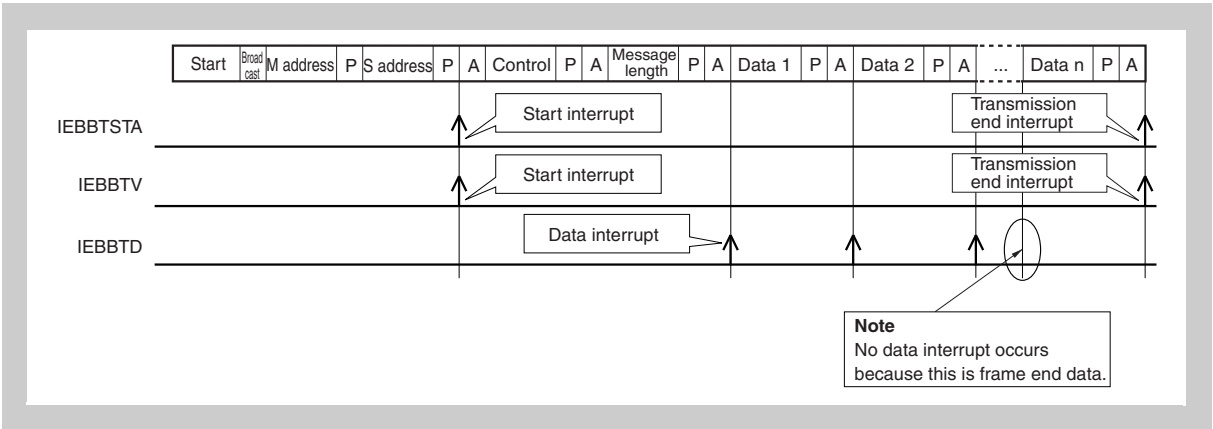


Figure 30-14 Interrupt occurrence timing

(3) Interrupt servicing examples

(a) Start interrupt CPU processing flow example

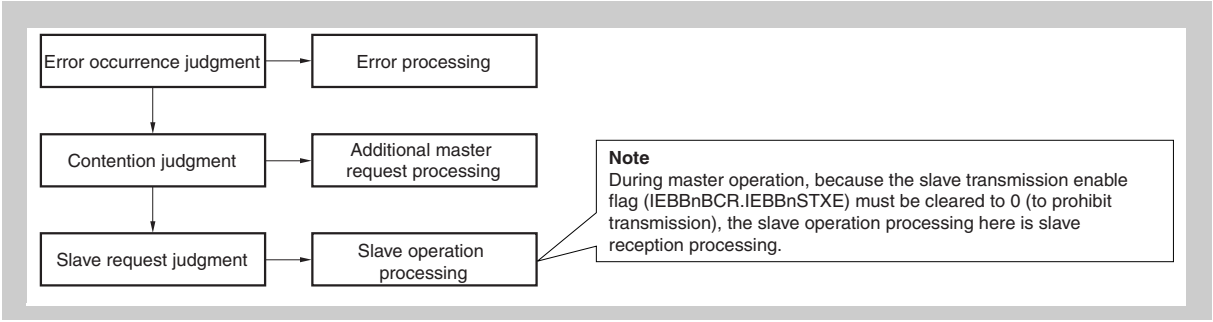


Figure 30-15 Start interrupt CPU processing flow example

(b) Transmission completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

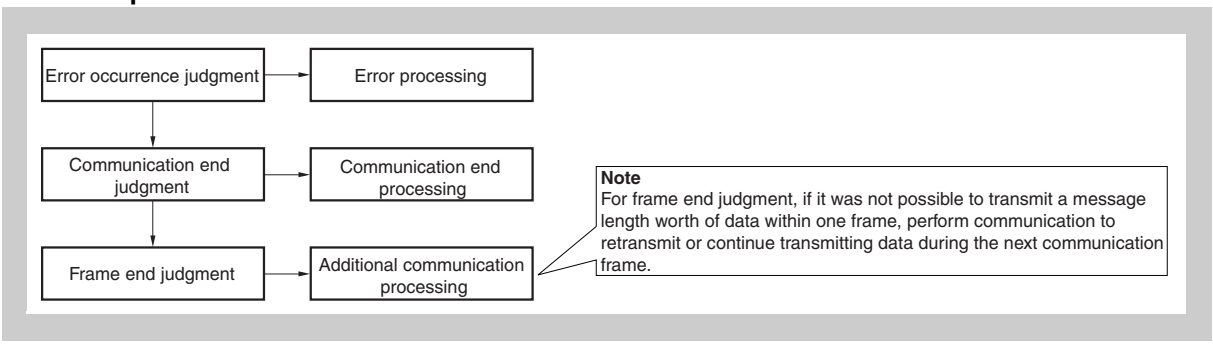


Figure 30-16 Transmission completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

(c) Data interrupt CPU processing example

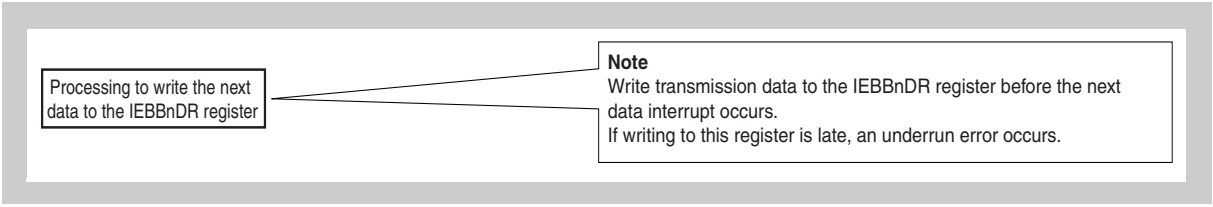


Figure 30-17 Data interrupt CPU processing

30.5.4 Master transmission (FIFO mode)

The unit transmits data and commands to the slave unit as the master.

Transmission data is written into the buffer in advance, and then a master request is issued.

(1) Register settings

After specifying the initial settings in *30.5.2 "Initial settings"*, set up the registers below before starting communication.

Table 30-46 Standard initial processing

Register name	Function	Example
IEBBnSAR	Communication partner unit address	102H
IEBBnCDR or IEBBnTCD	Control data (AH, BH, EH, FH)	FH
IEBBnTDL	Message length	02H
IEBBnDR	Data (all data to be transmitted)	11H, …

Table 30-47 Communication startup processing

Register name	Function	Example
IEBBnBCR	Communication startup processing	C8H

(2) Interrupt occurrence timing

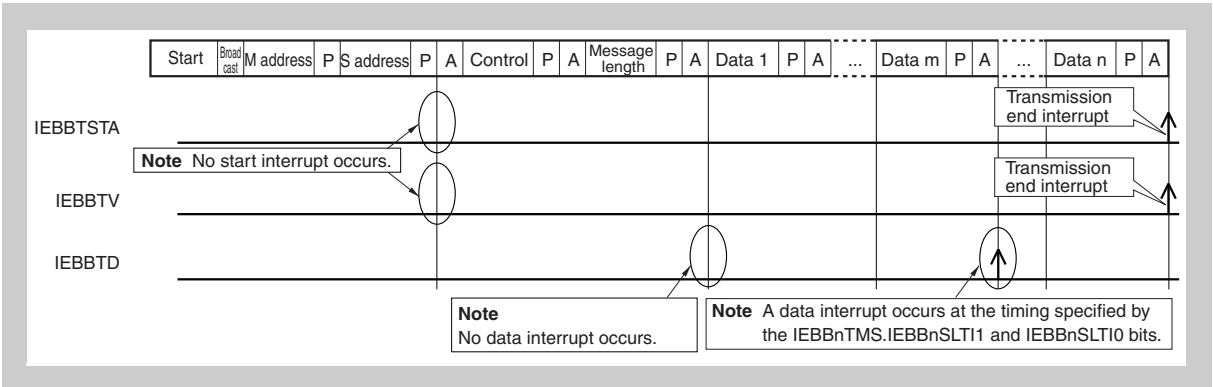


Figure 30-18 Interrupt occurrence timing

(3) Interrupt servicing examples

(a) Transmission completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

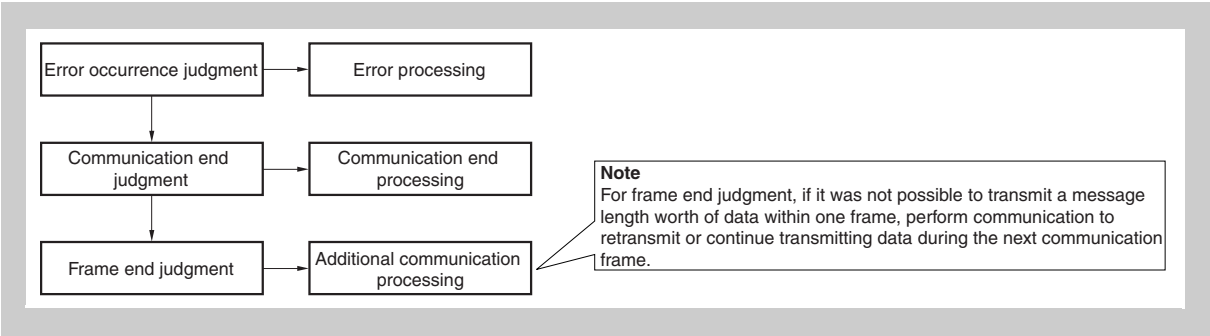


Figure 30-19 Transmission completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

30.5.5 Master reception (single mode)

The unit receives data and commands from the slave unit as the master. Because the slave transfers the message length field in the case of master reception, indicate the message length of data to be transmitted to the slave, such as during other communication. Read the received data one byte at a time by using data interrupts.

(1) Register settings

After specifying the initial settings in *30.5.2 "Initial settings"*, set up the registers below before starting communication.

Table 30-48 Standard initial processing

Register name	Function	Example
IEBBnSAR	Communication partner unit address	102H
IEBBnCDR or IEBBnTCD	Control data (0H, 3H, 4H, 5H, 6H, 7H)	7H

Table 30-49 Communication startup processing

Register name	Function	Example
IEBBnBCR	Communication startup processing	C8H

(2) Interrupt occurrence timing

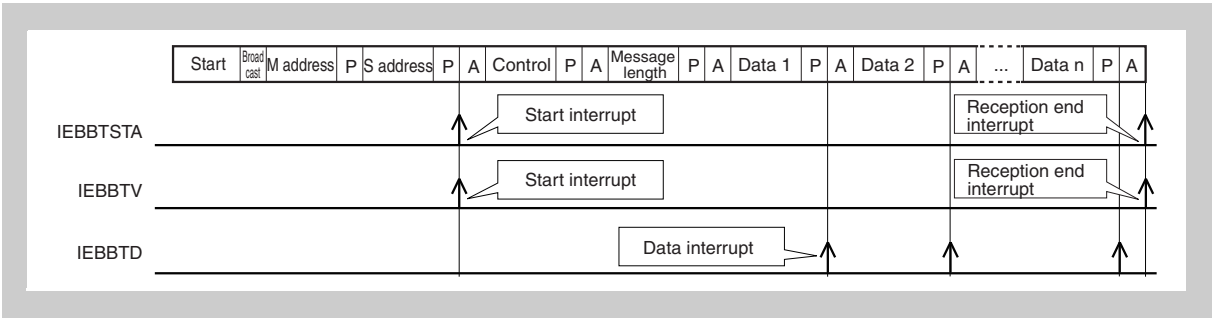


Figure 30-20 Interrupt occurrence timing

(3) Interrupt servicing examples

(a) Start interrupt CPU processing flow example

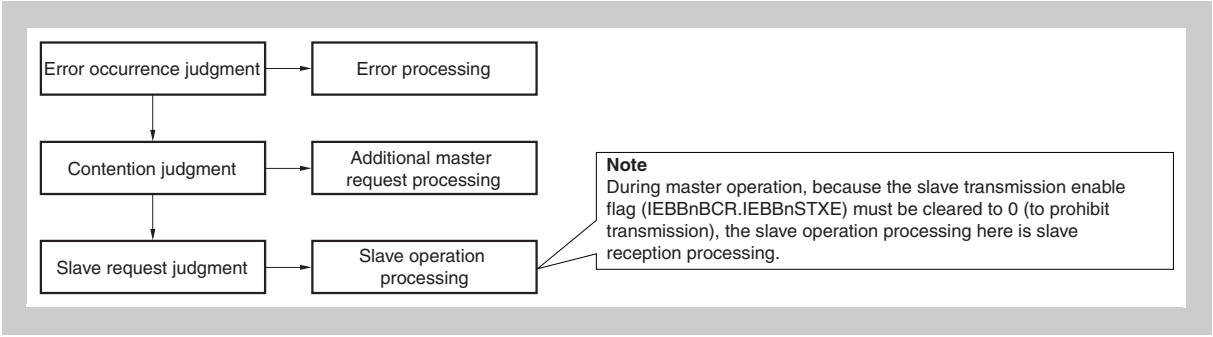


Figure 30-21 Start interrupt CPU processing flow example

(b) Reception completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

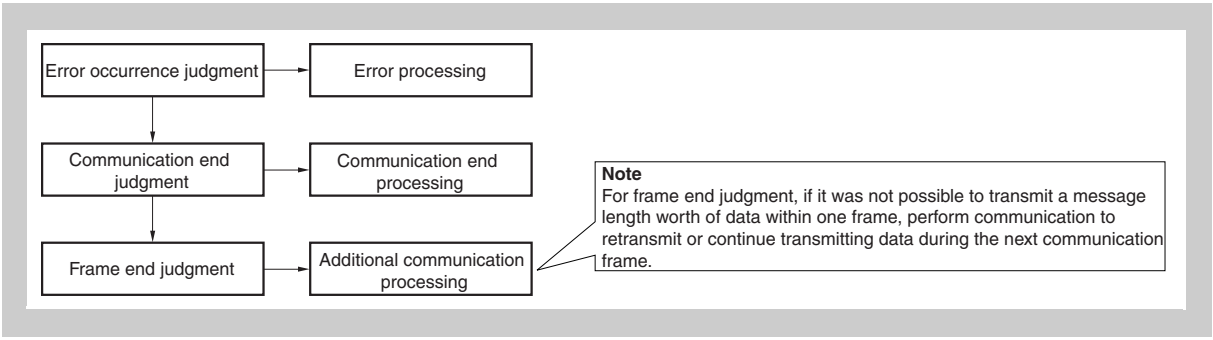


Figure 30-22 Reception completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

(c) Data interrupt CPU processing example

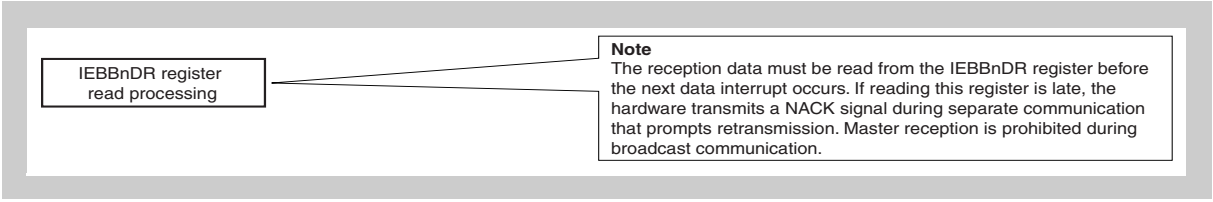


Figure 30-23 Data interrupt CPU processing

30.5.6 Master reception (FIFO mode)

The unit receives data and commands from the slave unit as the master. Because the slave transfers the message length field in the case of master reception, indicate the message length of data to be transmitted to the slave, such as during other communication.

(1) Register settings

After specifying the initial settings in *30.5.2 "Initial settings"*, set up the registers below before starting communication.

Table 30-50 Standard initial processing

Register name	Function	Example
IEBBnSAR	Communication partner unit address	102H
IEBBnCDR or IEBBnTCD	Control data (0H, 3H, 4H, 5H, 6H, 7H)	7H

Table 30-51 Communication startup processing

Register name	Function	Example
IEBBnBCR	Communication startup processing	C8H

(2) Interrupt occurrence timing

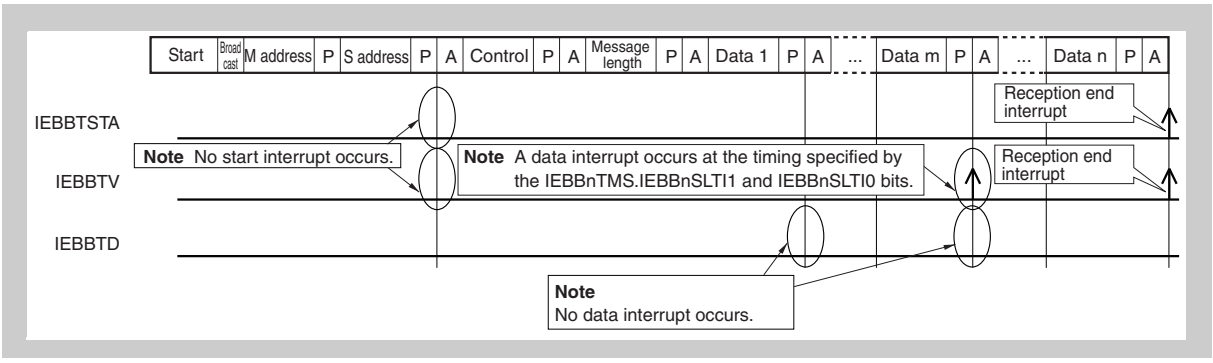


Figure 30-24 Interrupt occurrence timing

(3) Interrupt servicing examples

(a) Reception completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

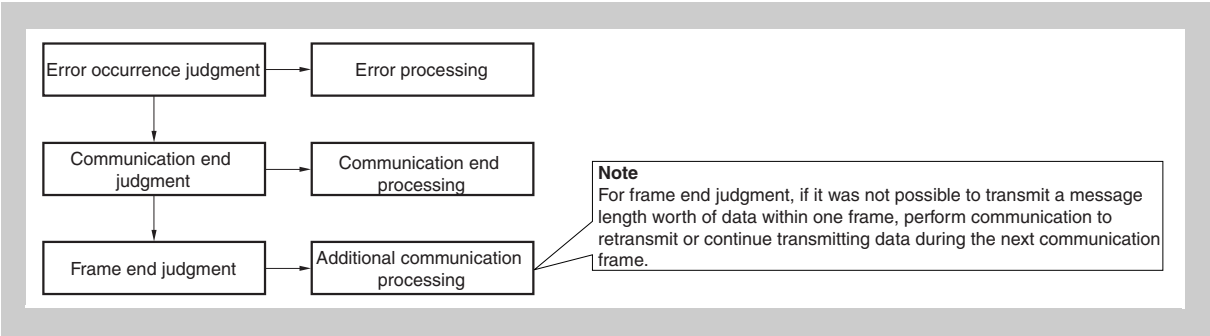


Figure 30-25 Reception completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

30.5.7 Slave transmission (single mode)

The unit transfers data and commands to the master unit as a slave.

Data interrupts are used to write transmission data to the IEBBnDR register for each one-byte transfer.

(1) Register settings

After specifying the initial settings in *30.5.2 "Initial settings"*, set up the registers below before starting communication.

Table 30-52 Standard initial processing

Register name	Function	Example
IEBBnDLR	Message length (other than during slave status transmission)	02H
IEBBnDR	Data (1st byte of data)	11H

Caution When starting slave transmission, information such as the value to be set to the message length register (IEBBnDLR) and which data is to be returned (the value to be set to the IEBBnDR register) must be assigned in advance by the master, such as during separate communication.

Table 30-53 Communication startup processing

Register name	Function	Example
IEBBnBCR	Communication startup processing	90H

(2) Interrupt occurrence timing

(a) When the control bit 3H or 7H, which is addressed to the unit, is received

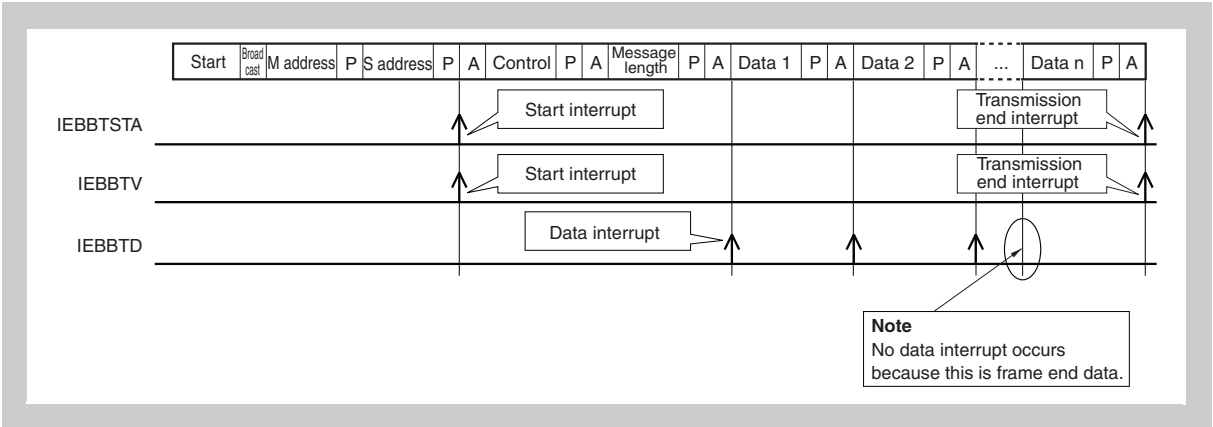


Figure 30-26 When the control bit 3H or 7H, which is addressed to the unit, is received

(b) When the control bit 0H or 6H, which is addressed to the unit, is received (or when 4H or 5H is received from the locked master while the unit is locked)

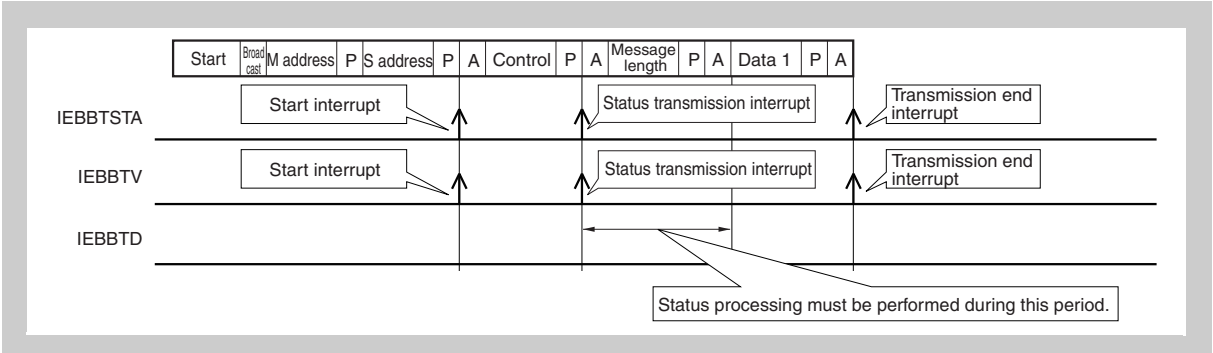


Figure 30-27 When the control bit 0H or 6H, which is addressed to the unit, is received (or when 4H or 5H is received from the locked master while the unit is locked)

(c) When the control bit 0H, 4H, or 5H, which is addressed to the unit, is received from a unit other than the locked master while the unit is locked

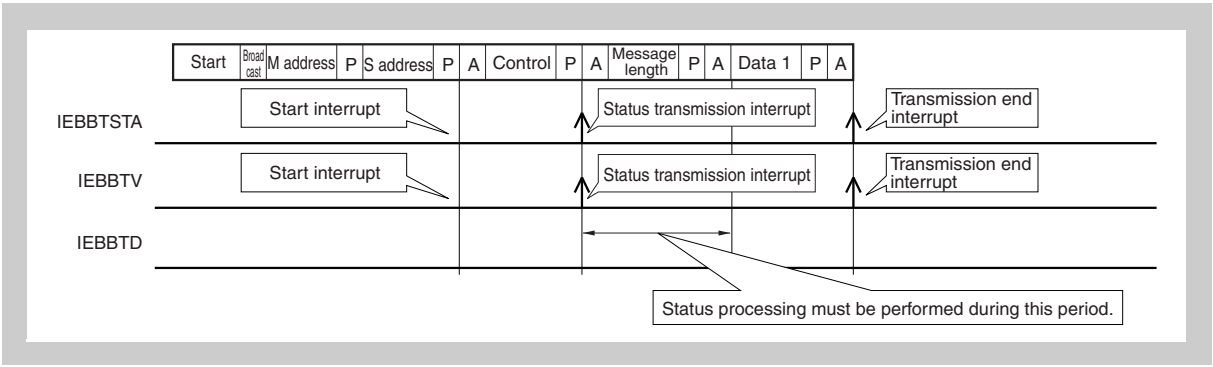


Figure 30-28 When the control bit 0H, 4H, or 5H, which is addressed to the unit, is received from a unit other than the locked master while the unit is locked

(3) Interrupt servicing examples

(a) Start interrupt CPU processing flow example

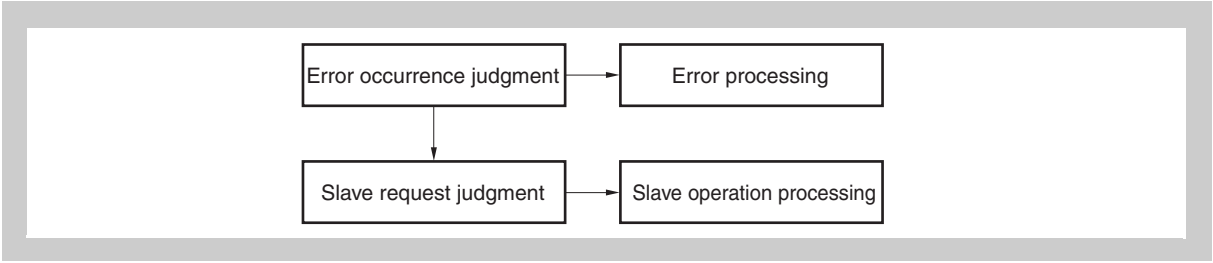


Figure 30-29 Start interrupt CPU processing flow example

(b) Transmission completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

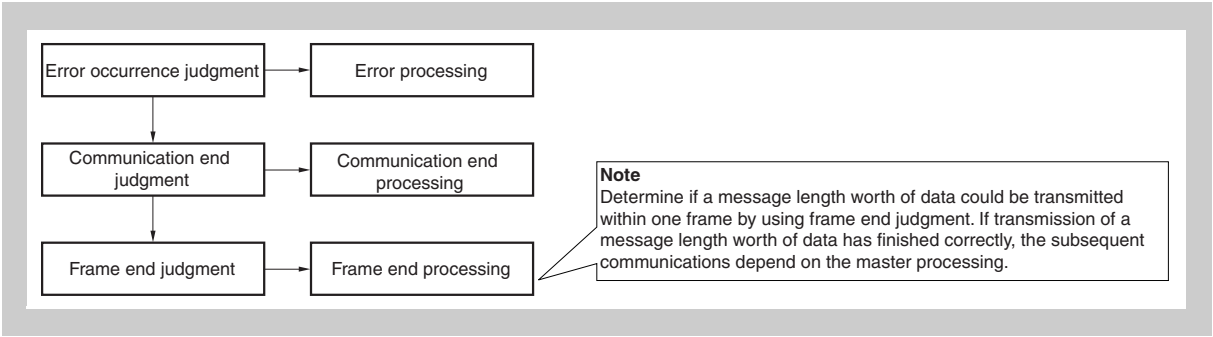


Figure 30-30 Transmission completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

(c) Save status transmission request processing flow example

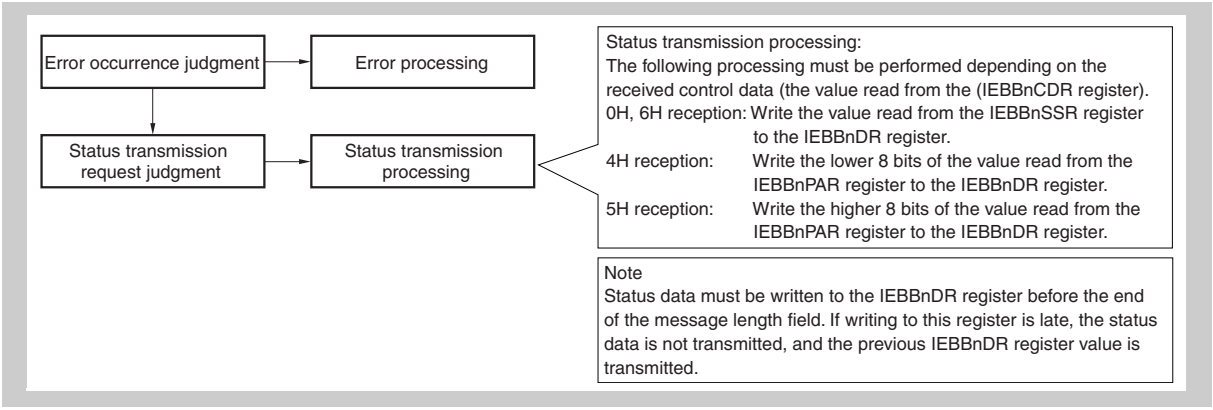


Figure 30-31 Slave status transmission request processing flow

(d) Data interrupt CPU processing example

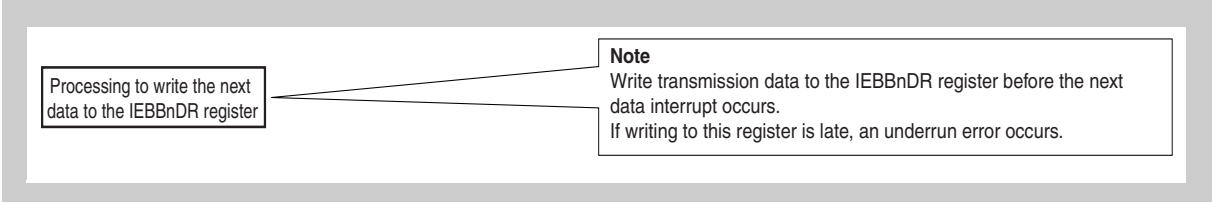


Figure 30-32 Data interrupt CPU processing

30.5.8 Slave transmission (FIFO mode)

The unit transfers data and commands to the master unit as a slave.

Write the transmission data to the buffer in advance.

(1) Register settings

After specifying the initial settings in *30.5.2 "Initial settings"*, set up the registers below before starting communication.

Table 30-54 Standard initial processing

Register name	Function	Example
IEBBnTDL	Message length (other than during slave status transmission)	02H
IEBBnDR	Data (all data to be transmitted)	11H, …

Caution When starting slave transmission, information such as the value to be set to the message length register (IEBBnTDL) and which data is to be returned (the value to be set to the IEBBnDR register) must be assigned in advance by the master, such as during separate communication.

Table 30-55 Communication startup processing

Register name	Function	Example
IEBBnBCR	Communication startup processing	90H

(2) Interrupt occurrence timing

(a) When the control bit 3H or 7H, which is addressed to the unit, is received

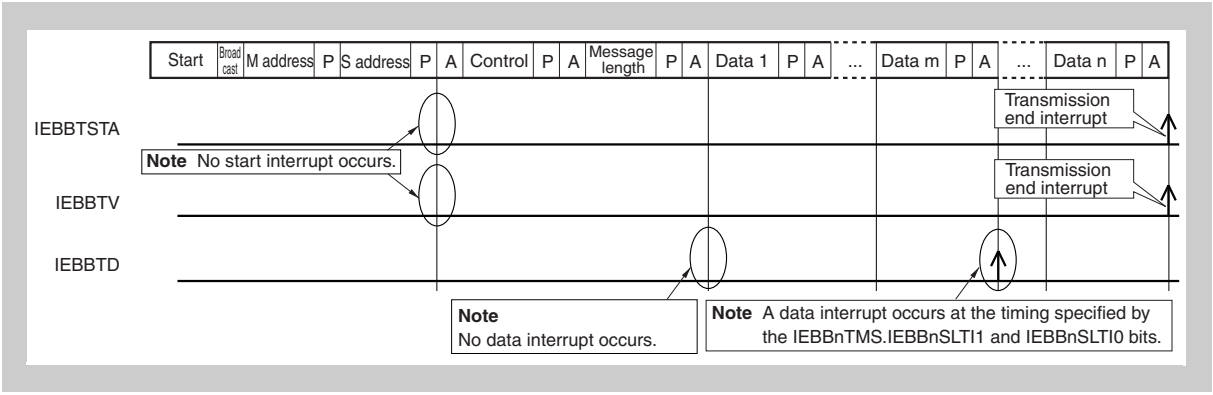


Figure 30-33 When the control bit 3H or 7H, which is addressed to the unit, is received

(b) When the control bit 0H or 6H, which is addressed to the unit, is received (or when 4H or 5H is received from the locked master while the unit is locked)

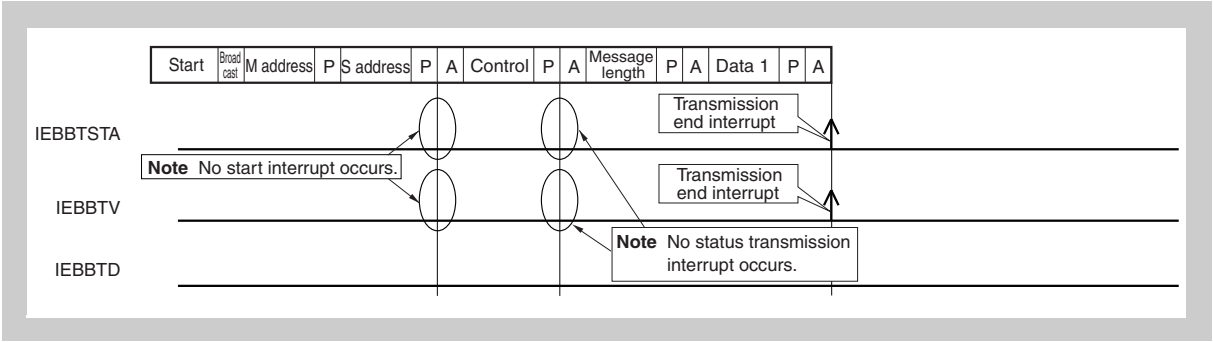


Figure 30-34 When the control bit 0H or 6H, which is addressed to the unit, is received (or when 4H or 5H is received from the locked master while the unit is locked)

(c) When the control bit 0H, 4H, or 5H, which is addressed to the unit, is received from a unit other than the locked master while the unit is locked

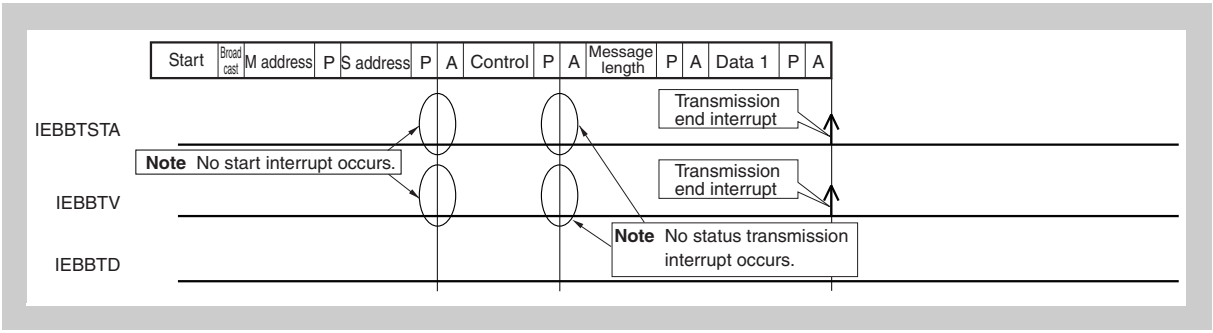


Figure 30-35 When the control bit 0H, 4H, or 5H, which is addressed to the unit, is received from a unit other than the locked master while the unit is locked

(3) Interrupt servicing examples

(a) Transmission completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

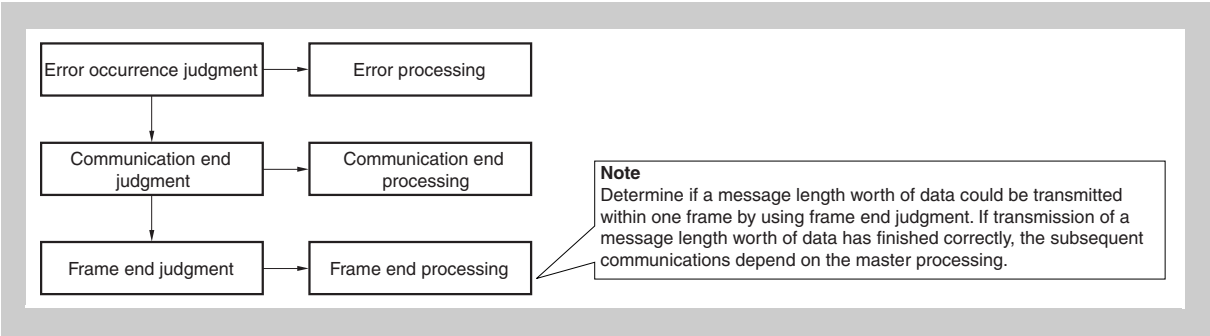


Figure 30-36 Transmission completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

(b) Slave status transmission request processing flow example

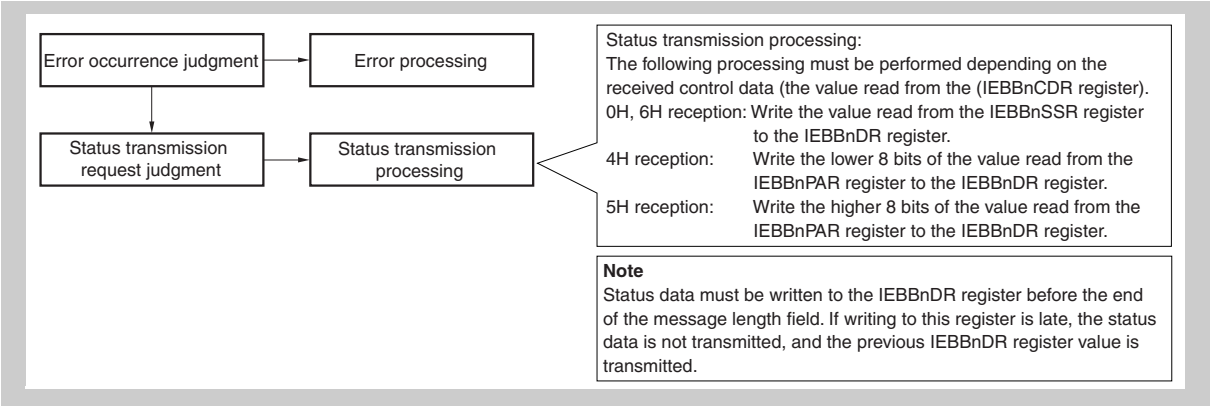


Figure 30-37 Slave status transmission request processing flow

In the FIFO mode, if the unit loses a conflict between a unit master transmission request and a slave transmission request addressed to the unit, because the slave transmission enable flag (IEBBnBCR.IEBBnSTXE) is not set (to 1) for the unit, a NACK signal is transmitted using the control data field and communication ends. Next, specify the slave transmission data for the FIFO buffer, set the slave transmission enable flag (to 1), and prepare to receive another slave transmission request from the master.

30.5.9 Slave reception (single mode)

The unit receives data and commands from the master unit as a slave.

Read the received data one byte at a time by using data interrupts.

(1) Register settings

After specifying the initial settings in *30.5.2 "Initial settings"*, set up the registers below before starting communication.

Table 30-56 Communication startup processing

Register name	Function	Example
IEBBnBCR	Communication startup processing	88H

(2) Interrupt occurrence timing

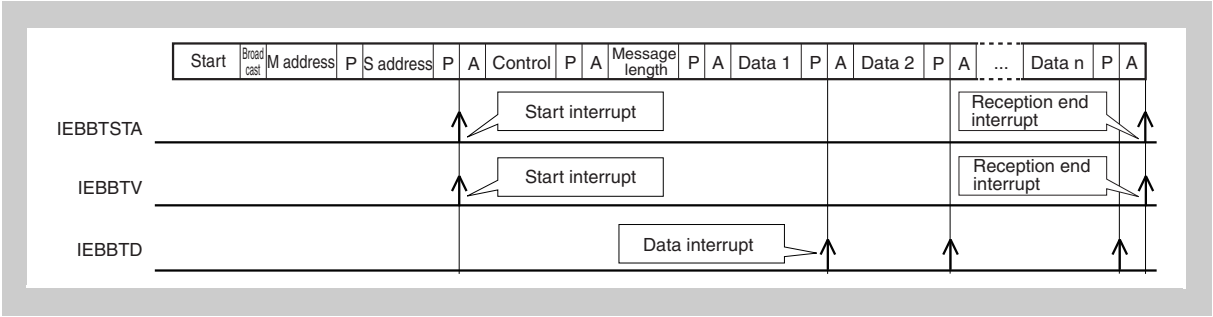


Figure 30-38 Interrupt occurrence timing

(3) Interrupt servicing examples

(a) Start interrupt CPU processing flow example

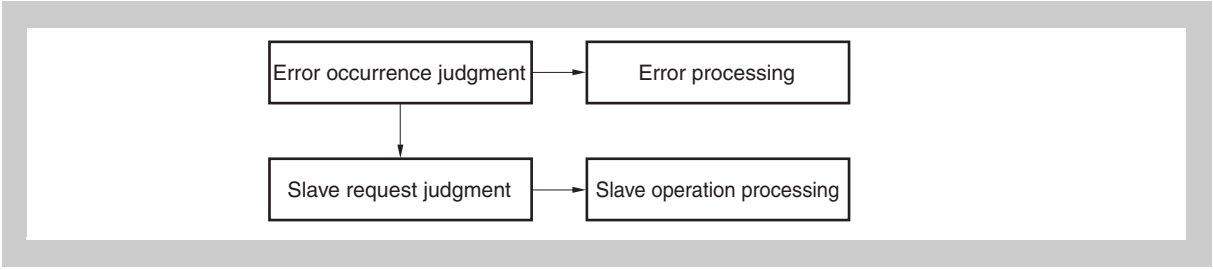


Figure 30-39 Start interrupt CPU processing flow example

(b) Reception completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

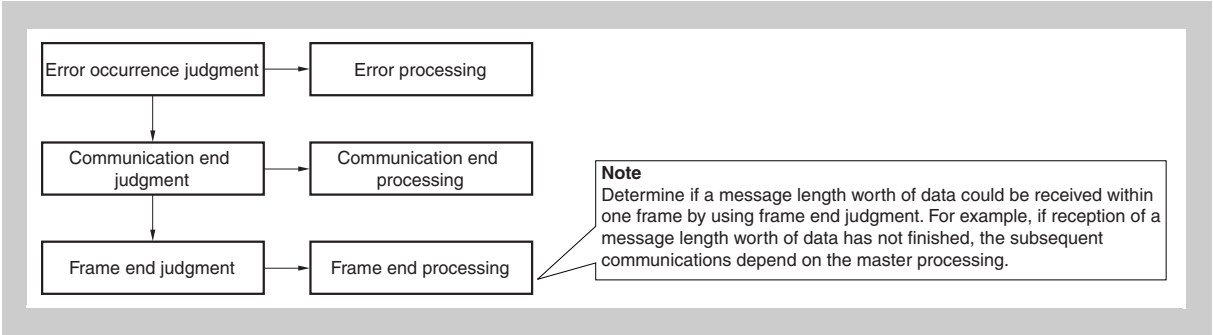


Figure 30-40 Reception completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

(c) Data interrupt CPU processing example

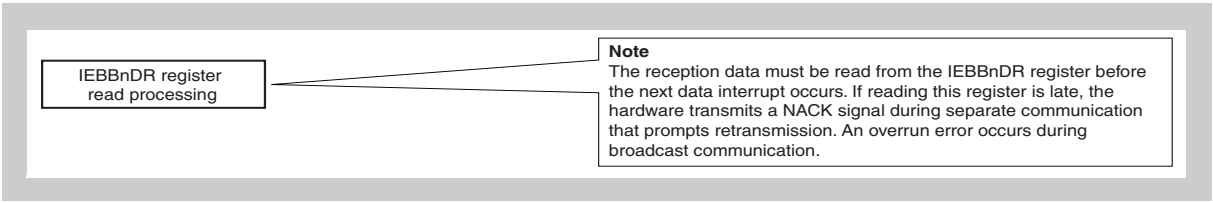


Figure 30-41 Data interrupt CPU processing

30.5.10 Slave reception (FIFO mode)

The unit receives data and commands from the master unit as a slave.

(1) Register settings

After specifying the initial settings in *30.5.2 "Initial settings"*, set up the registers below before starting communication.

Table 30-57 Communication startup processing

Register name	Function	Example
IEBBnBCR	Communication startup processing	88H

(2) Interrupt occurrence timing

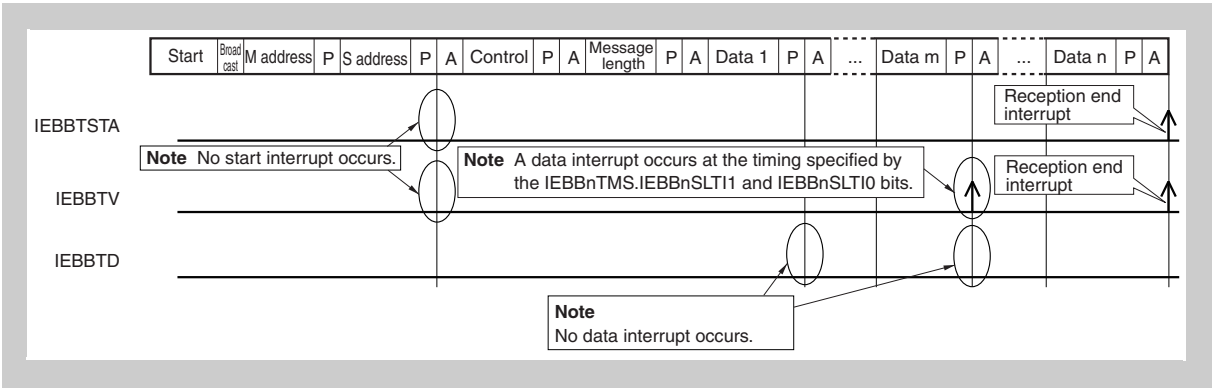


Figure 30-42 Interrupt occurrence timing

(3) Interrupt servicing examples

(a) Reception completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

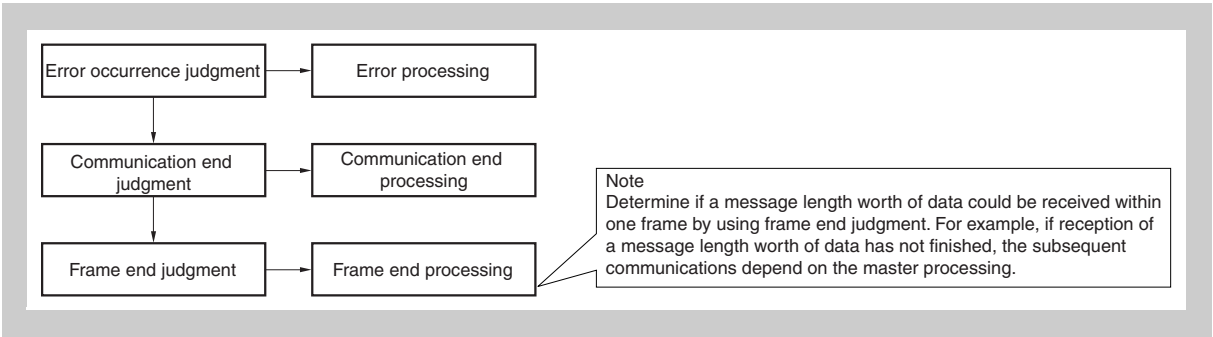


Figure 30-43 Reception completion interrupt (IEBBTV, IEBBTSTA) CPU processing flow example

30.6 Setup Procedures

30.6.1 Master transmission (single mode)

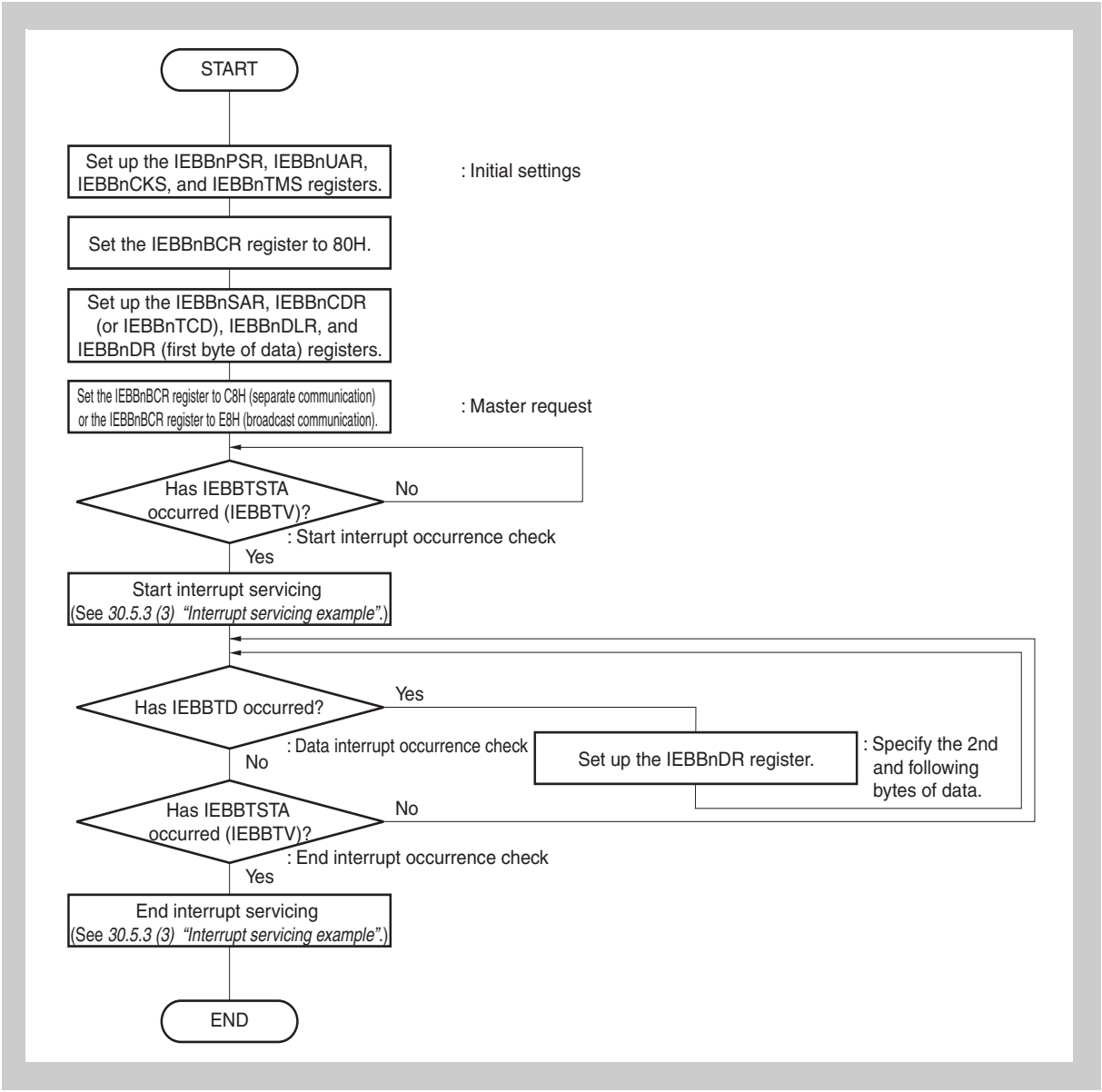


Figure 30-44 Master transmission (single mode)

30.6.2 Master transmission (FIFO mode)

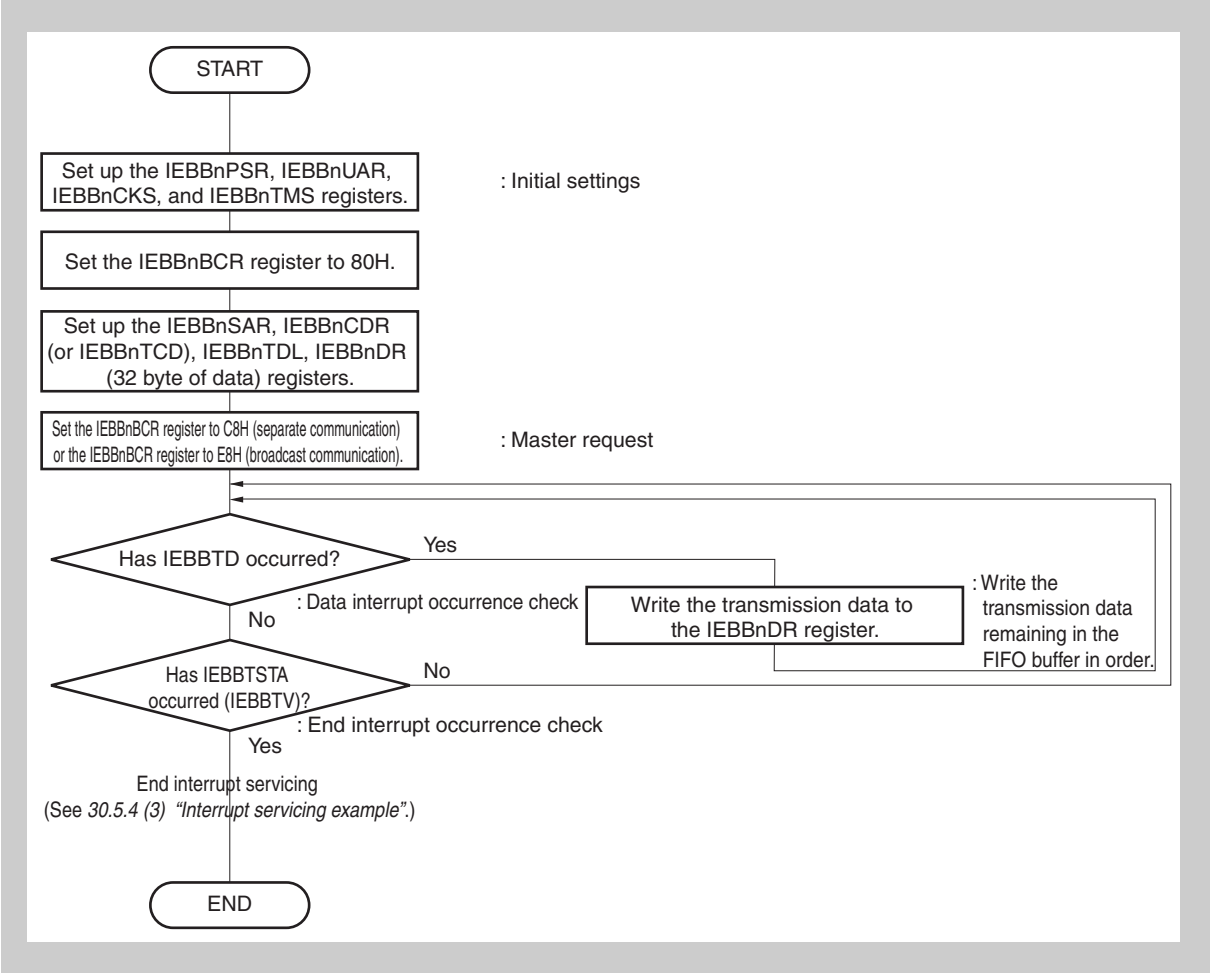


Figure 30-45 Master transmission (FIFO mode)

30.6.3 Master reception (single mode)

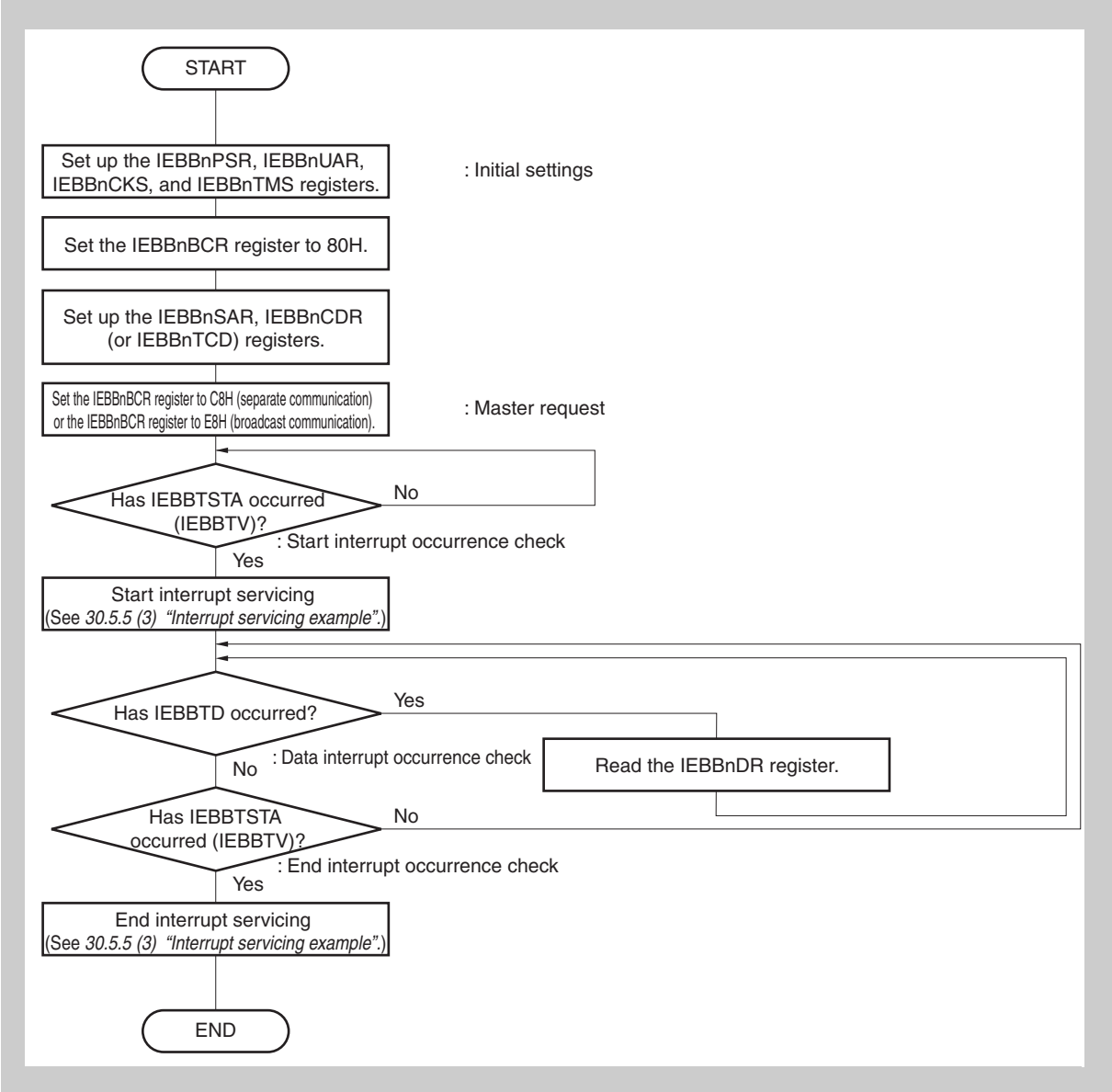


Figure 30-46 Master reception (single mode)

30.6.4 Master reception (FIFO mode)

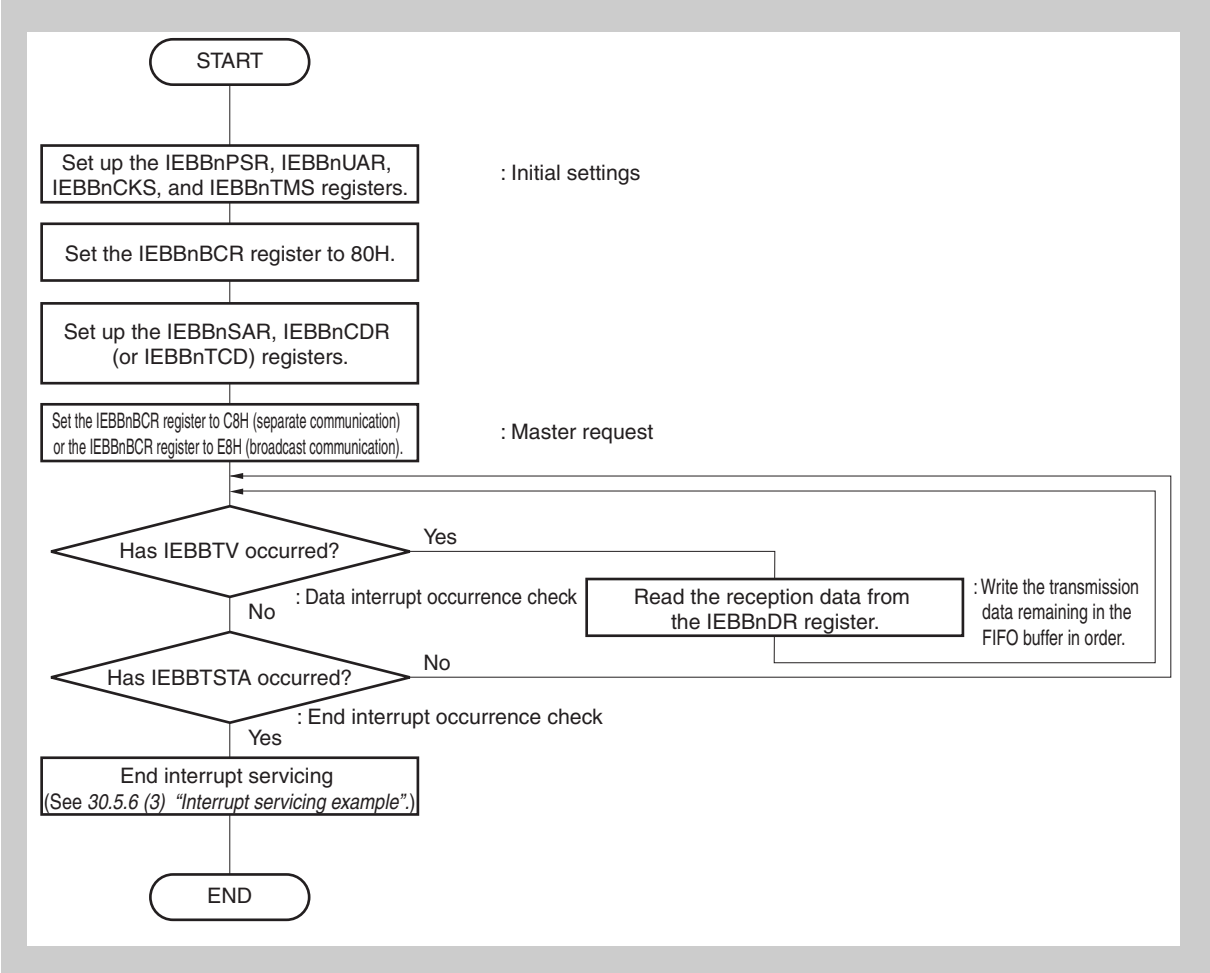


Figure 30-47 Master reception (FIFO mode)

30.6.5 Slave transmission (single mode)

(1) When the control bit 3H or 7H is received

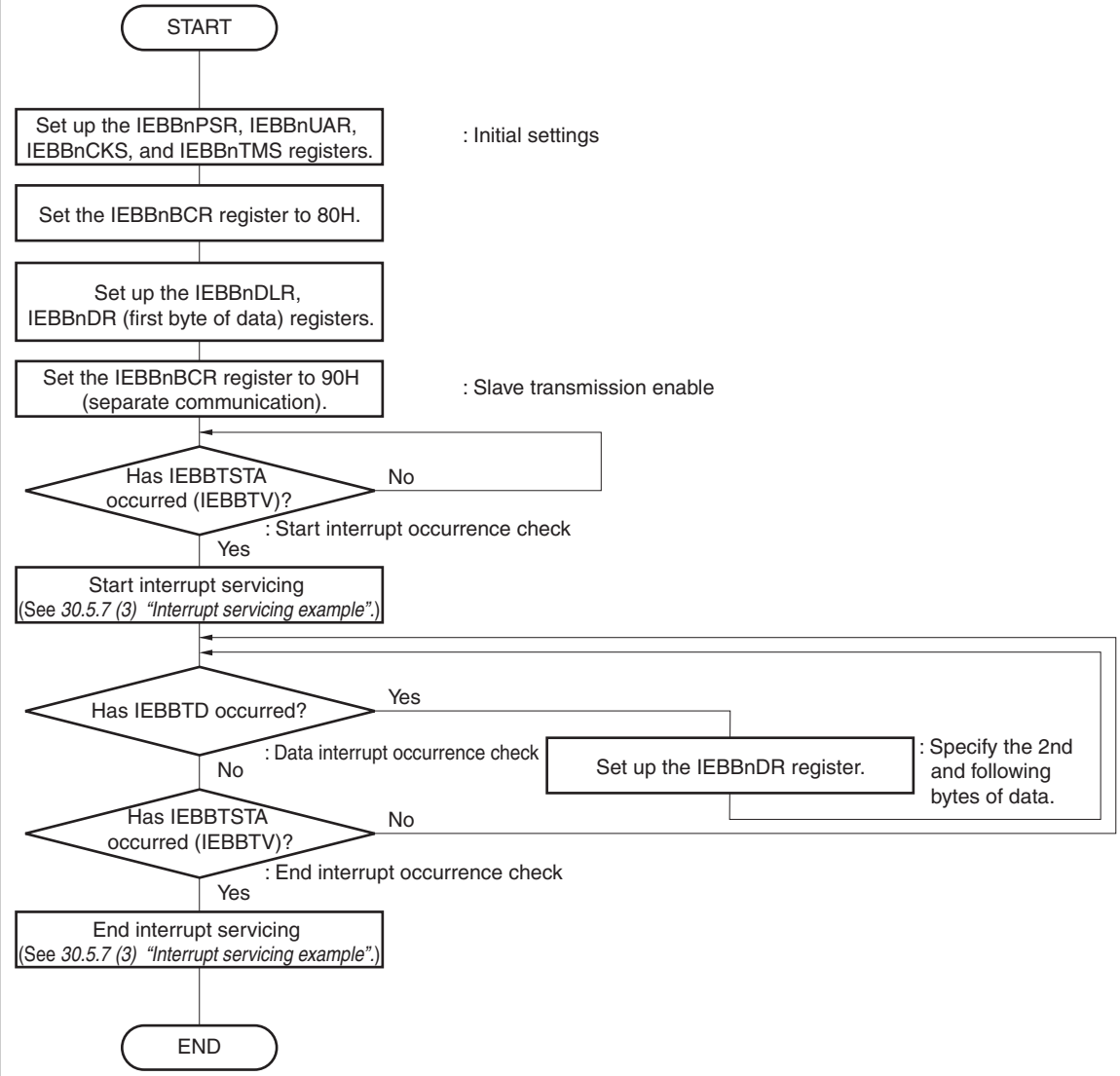


Figure 30-48 Slave transmission (single mode): When the control bit 3H or 7H is received

(2) When the control bit 0H or 6H is received (or when 4H or 5H is received from the locked master while the unit is locked)

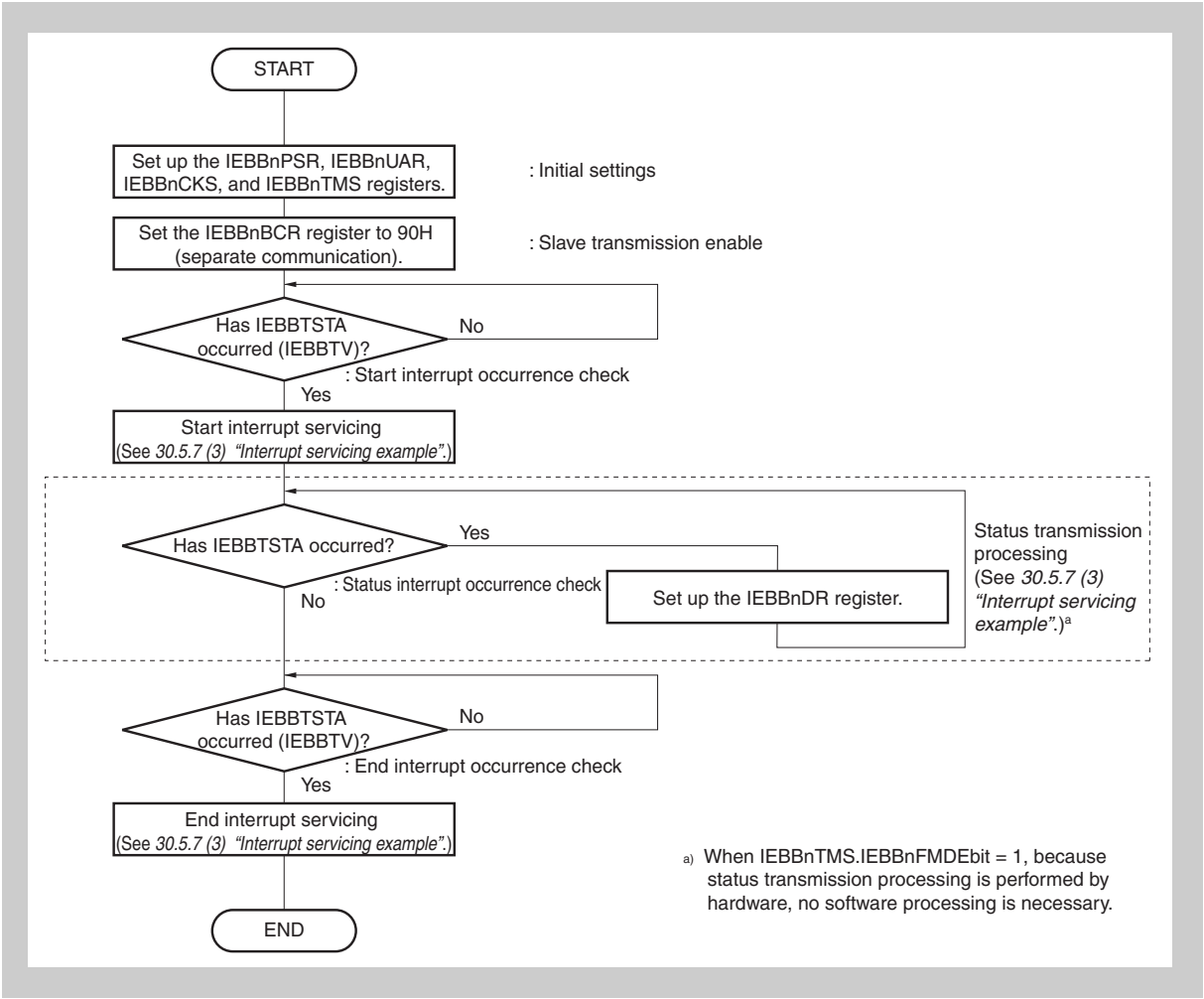


Figure 30-49 Slave transmission (single mode): When the control bit 0H or 6H is received (or when 4H or 5H is received from the locked master while the unit is locked)

(3) When the control bit 0H, 4H, or 5H, which is addressed to the unit, is received from a unit other than the locked master while the unit is locked

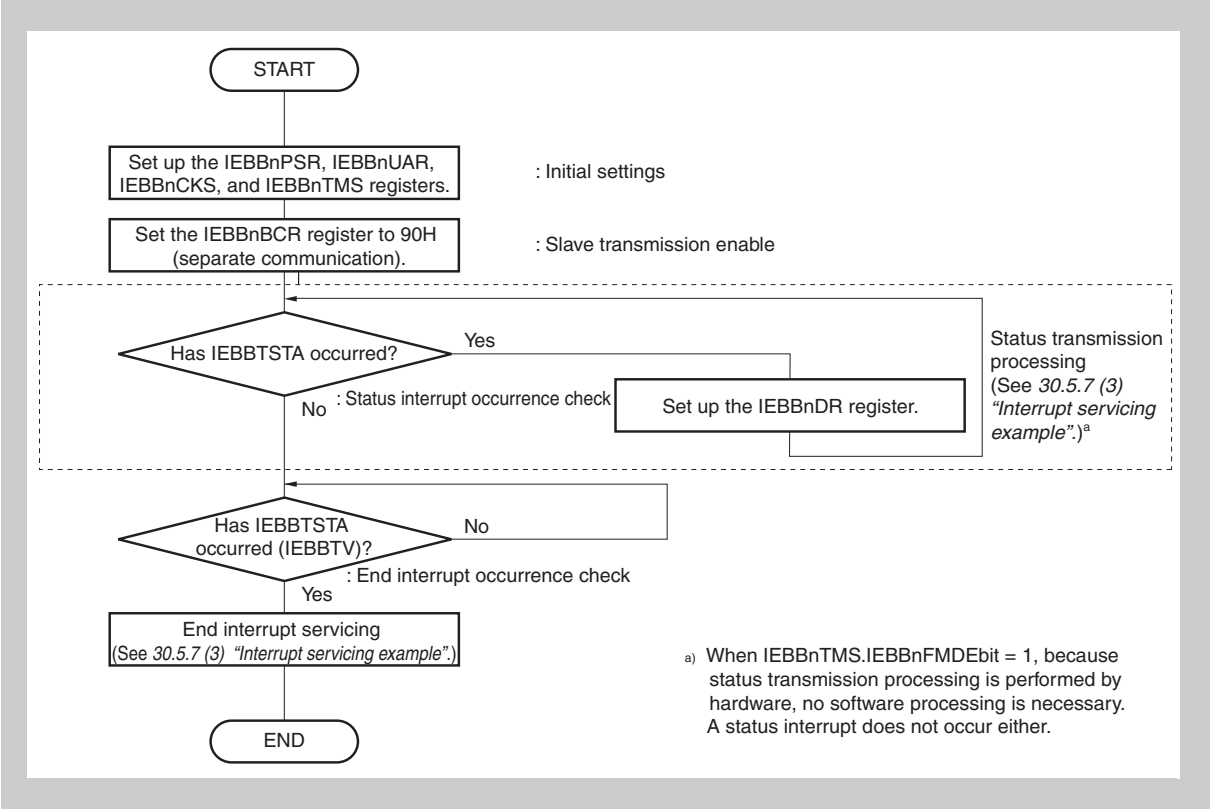


Figure 30-50 Slave transmission (single mode): When the control bit 0H, 4H, or 5H, which is addressed to the unit, is received from a unit other than the locked master while the unit is locked

30.6.6 Slave transmission (FIFO mode)

(1) When the control bit 3H or 7H is received

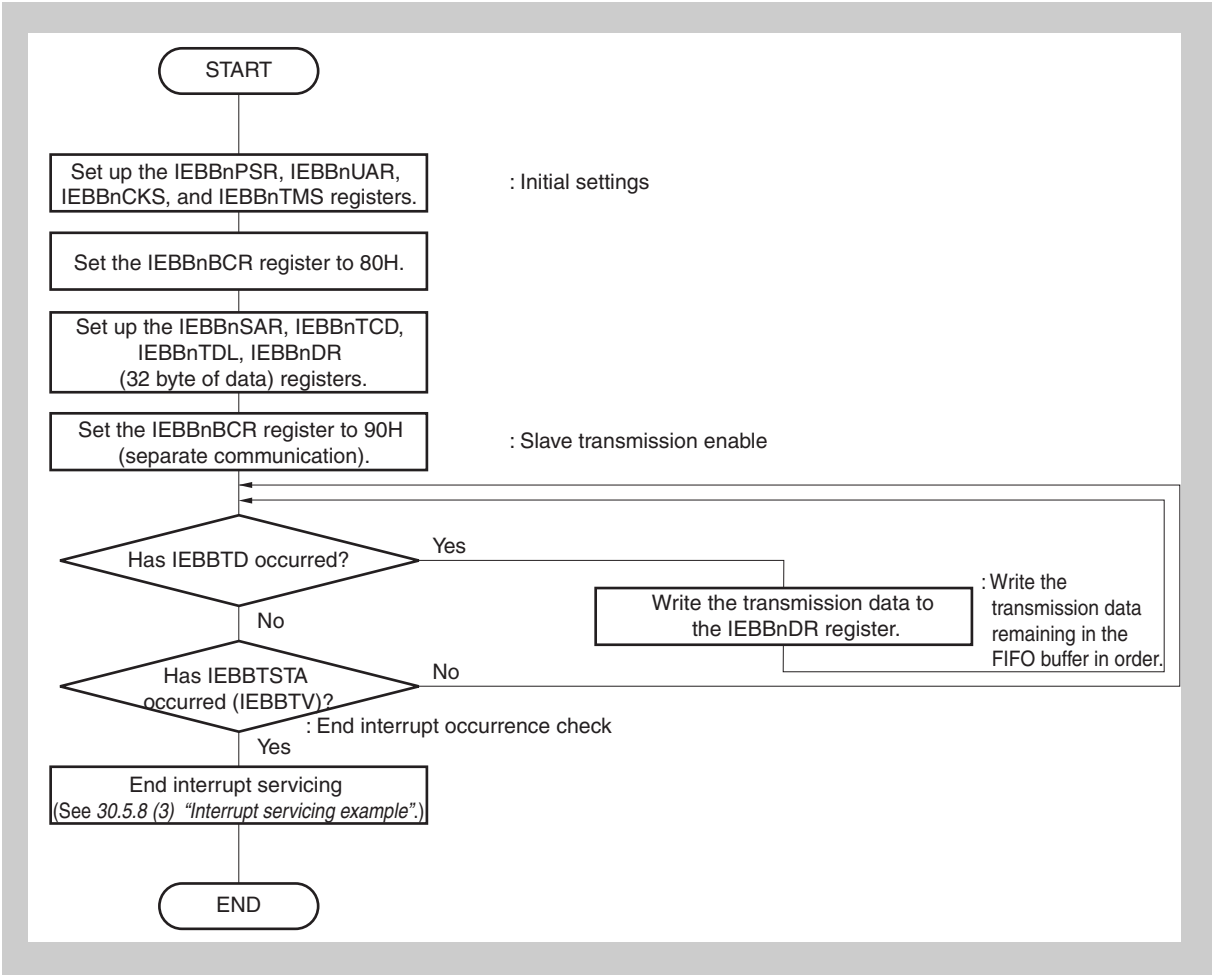


Figure 30-51 Slave transmission (FIFO mode): When the control bit 3H or 7H is received

(2) When the control bit 0H or 6H is received (or when 4H or 5H is received from the locked master while the unit is locked)

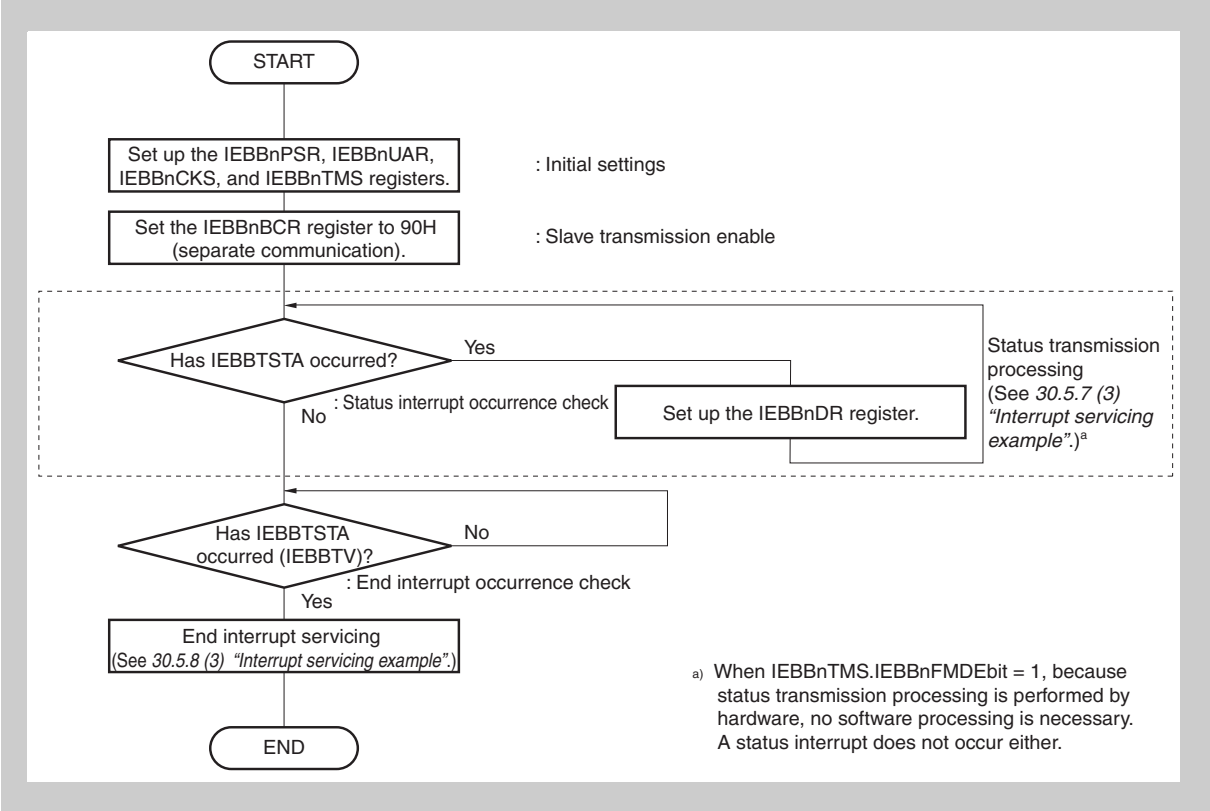


Figure 30-52 Slave transmission (FIFO mode): When the control bit 0H or 6H is received (or when 4H or 5H is received from the locked master while the unit is locked)

(3) When the control bit 0H, 4H, or 5H, which is addressed to the unit, is received from a unit other than the locked master while the unit is locked

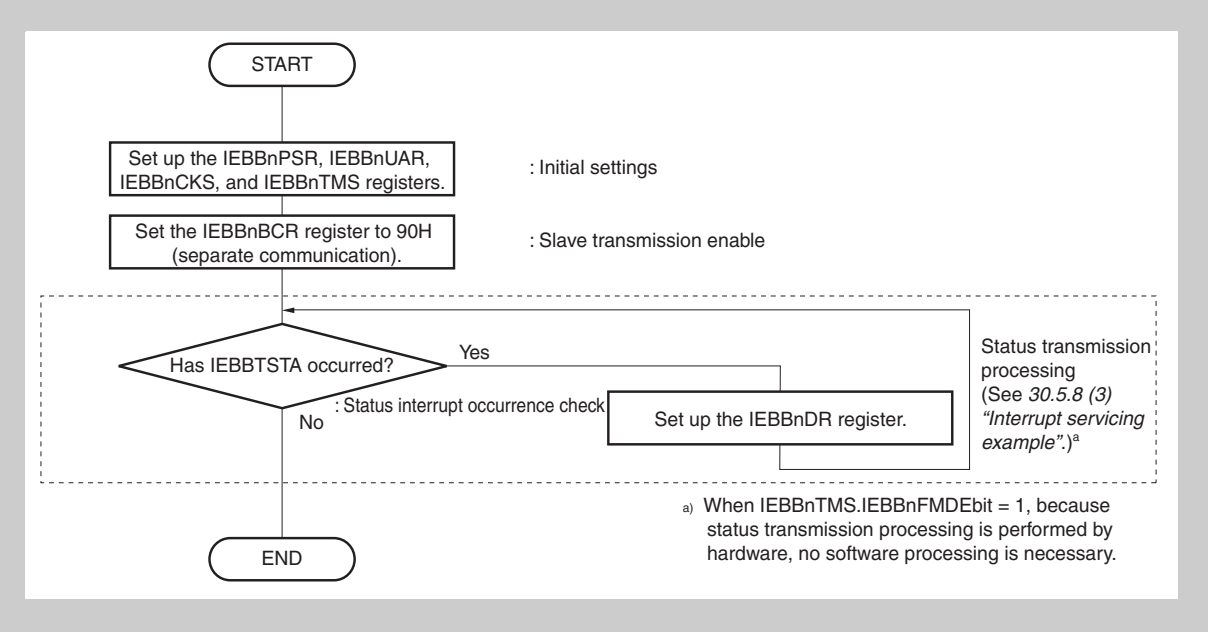


Figure 30-53 Slave transmission (FIFO mode): When the control bit 0H, 4H, or 5H, which is addressed to the unit, is received from a unit other than the locked master while the unit is locked

30.6.7 Slave reception (single mode)

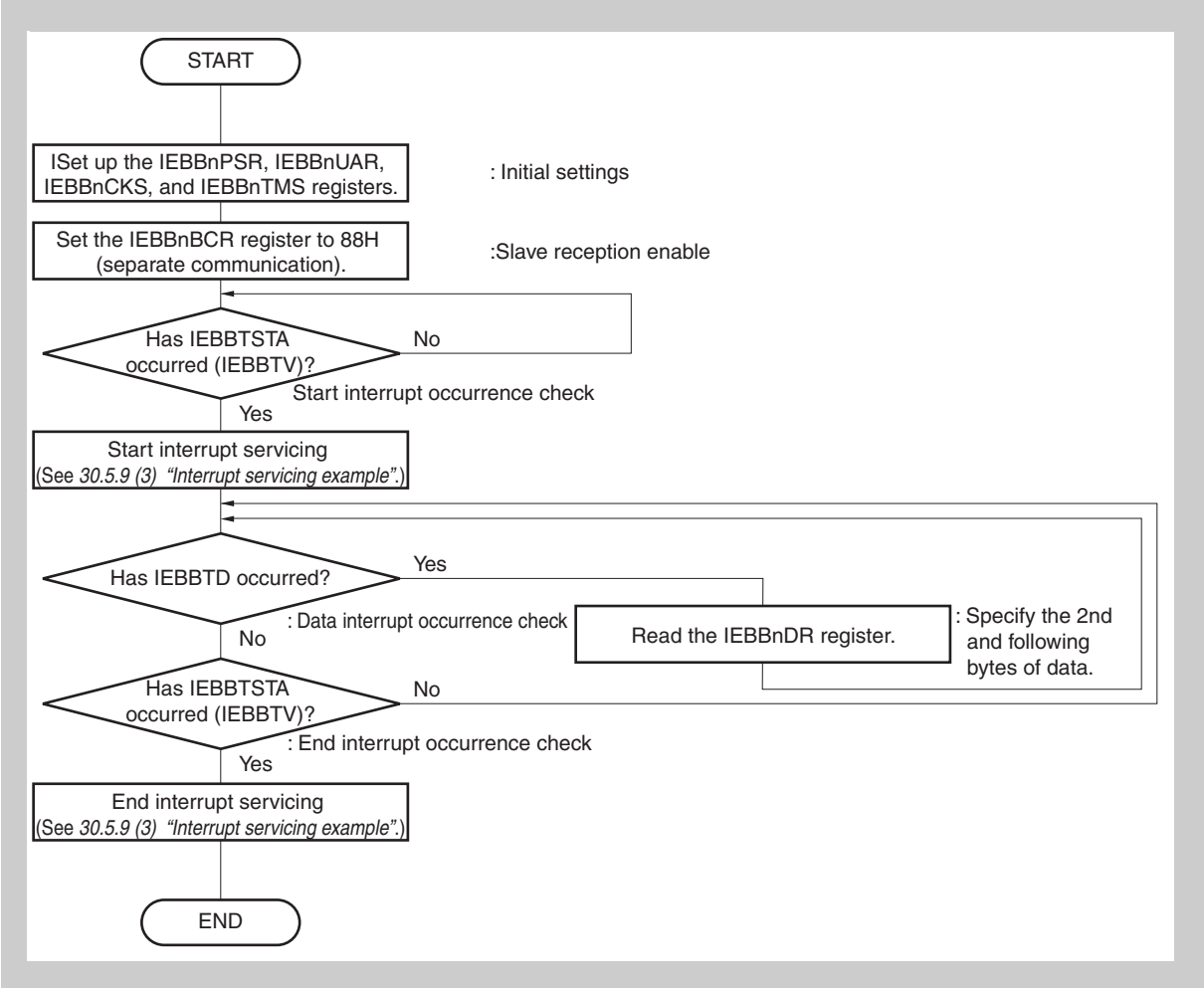


Figure 30-54 Slave reception (single mode)

30.6.8 Slave reception (FIFO mode)

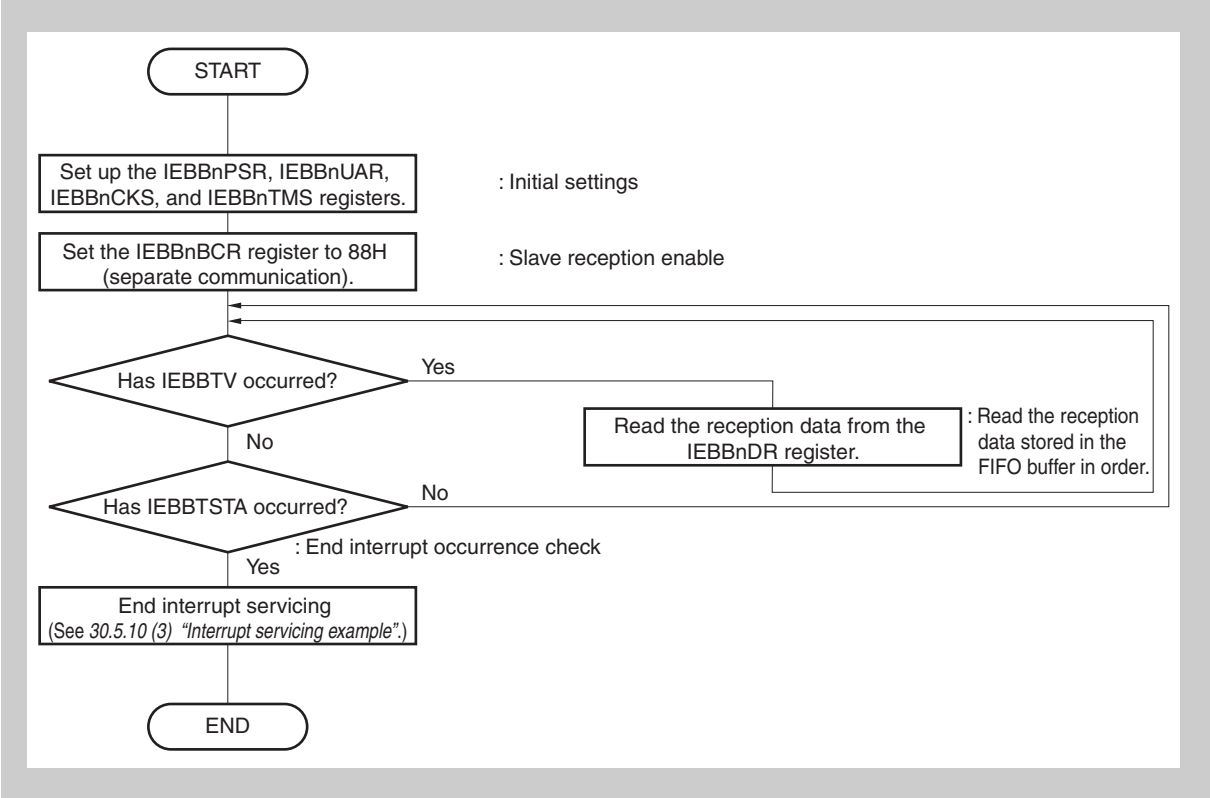


Figure 30-55 Slave reception (FIFO mode)

30.7 Functions

30.7.1 IEBus communication protocol

The communication protocol of the IEBus is as follows.

(1) Multi-task mode

All the units connected to the IEBus can transfer data to the other units.

(2) Broadcast communication

Communication between one unit and multiple units can be performed as follows.

- Group broadcast communication: Broadcast communication to group units
- All-unit broadcast communication: Broadcast communication to all units

(3) Effective transmission speed

The effective transfer rate is in mode 1 or mode 2. (The V850E2/Sx4-H does not support mode 0 for the effective transfer rate.)

- Mode 1: Approx. 17 kbps
- Mode 2: Approx. 26 kbps

Caution Different modes (mode 1, mode 2) must not be mixed on one IEBus.

(4) Communication mode

Data transfer is executed in half-duplex asynchronous communication mode.

(5) Access control: CSMA/CD (Carrier Sense Multiple Access with Collision Detection)

The priority of the IEBus is as follows:

1. Broadcast communication takes precedence over individual communication (communication from one unit to another).
2. The lower master address takes precedence.

(6) Communication scale

The communication scale of IEBus is as follows:

- Number of units: 50 maximum
- Cable length: 150 m maximum (when twisted pair cable is used)

Caution The communication scale in an actual system differs depending on the characteristics of the cables, etc., constituting the IEBus driver/receiver and IEBus.

30.7.2 Determination of bus mastership (arbitration)

An operation to occupy the bus is performed when a unit connected to the IEBus controls the other units. This operation is called arbitration.

When multiple units simultaneously start transmission, arbitration is used to grant one of the units permission to occupy the bus.

Because only one unit is granted bus mastership as a result of arbitration, the priority conditions of the bus are predetermined as follows.

Caution Bus mastership is canceled if communication is aborted.

(1) Priority by communication type

Broadcast communication (communication from one unit to multiple units) takes precedence over normal communication (communication from one unit to another).

(2) Priority by master address

If the communication type is the same, communication with the lower master address takes precedence.

A master address consists of 12 bits, with unit 000H having the highest priority and unit FFFH having the lowest priority.

30.7.3 Communication mode

The IEBus has three communication modes, each of which has a different transfer rate. The V850E2/Sx4-H supports communication modes 1 and 2. The transfer rate and the maximum number of transfer bytes per communication frame in communication modes 1 and 2 are shown below.

Table 30-58 Transfer rate and maximum number of transfer bytes in each communication mode

Communication mode	Maximum number of transfer bytes (bytes/frame)	Effective transfer rate[a]
1	32 bytes/frame	Approx. 17 kbps
2	128 bytes/frame	Approx. 26 kbps

a) Effective transfer rate when the maximum number of transfer bytes is transmitted

Select the communication mode for each unit connected to the IEBus before starting communication. If the communication mode of the master unit and that of the partner unit (slave unit) are not the same, communication is not correctly executed.

30.7.4 Communication address

For the IEBus, each unit is assigned a specific 12-bit address. This communication address consists of the following identification numbers.

- Higher 4 bits: Group number (number to identify the group to which each unit belongs)
- Lower 8 bits: Unit number (number to identify each unit in a group)

30.7.5 Broadcast communication

Normally, transmission or reception is performed between the master unit and its partner slave unit on a one-to-one basis. During broadcast communication, however, multiple slave units exist and the master unit executes transmission to these slave units. Because multiple slave units exist, the NACK signal is returned by the communicating slave unit as an acknowledge bit.

Whether broadcast communication or normal communication is to be executed is selected by the broadcast bit. (For details about this bit, see *30.7.6 (2) "Broadcast bit"*.)

Broadcast communication is classified into two types: group-unit broadcast communication and all-unit broadcast communication. Group-unit broadcast and all-unit broadcast are identified by the value of the slave address. (For the slave address, see *30.7.6 (4) "Slave address field"*.)

(1) Group-unit broadcast communication

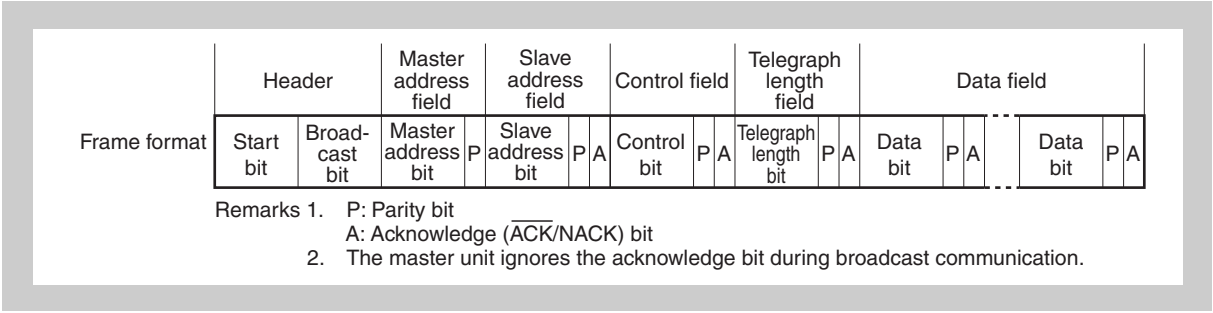
Broadcast communication is performed to the units in a group identified by the group number indicated by the higher 4 bits of the communication address.

(2) All-unit broadcast communication

Broadcast communication is performed to all the units, regardless of the value of the group number.

30.7.6 IEBus transfer format

The IEBus transfer signal format is shown in *Figure 30-56*.



Remarks 1. P: Parity bit
A: Acknowledge ($\overline{\text{ACK}}$/NACK) bit
2. The master unit ignores the acknowledge bit during broadcast communication.

Figure 30-56 IEBus transfer signal format

(1) Start bit

The start bit is a signal that informs the other units of the start of a data transfer.

The unit that is to start a data transfer outputs a high-level signal (start bit) from the IEBBTTXB pin for a specific time, and then starts outputting the broadcast bit.

If another unit has already output its start bit when one unit is to output the start bit, this unit does not output the start bit and instead waits for completion of output of the start bit by the other unit. When the output of the start bit by the other unit is complete, the unit starts outputting the broadcast bit in synchronization with the completion of the start bit output by the other unit.

The units other than the one that started communication detect this start bit, and enter the reception status.

(2) Broadcast bit

This bit indicates whether the master selects one slave (individual communication) or multiple slaves (broadcast communication) as the other party of communication.

When the broadcast bit is 0, it indicates broadcast communication. When it is 1, individual communication is indicated. Broadcast communication is classified into two types: group-unit communication and all-unit communication. These communication types are identified by the value of the slave address. (For the slave address, see *30.7.6 (4) "Slave address field"*.)

Because multiple slave units exist as a partner slave unit of communication in the case of broadcast communication, the NACK signal is returned as an acknowledge bit in each field subsequent to the master address field.

If multiple units start transmitting a communication frame at the same time, broadcast communication takes precedence over individual communication, and wins in arbitration.

If one unit occupies the bus as the master, the value set to the broadcast request flag (the IEBBnBCR.IEBBnALRQ bit) is output.

(3) Master address field

The master address field is output by the master to inform a slave of the master's address.

The configuration of the master address field is shown in *Figure 30-57*.

If multiple units start transmitting the broadcast bit at the same time, the master address field makes a judgment of arbitration.

The master address field compares the data it outputs with the data on the bus each time it has output one bit. If the master address output by the master address field is found to differ from the data on the bus as a result of comparison, it is assumed that the master has lost arbitration.

As a result, the master stops transmission and enters the reception status. Because the IEBus is configured of wired AND, the unit having the smallest master address of the units participating in arbitration (arbitration masters) wins arbitration.

After a 12-bit master address has been output, only one unit remains in the transmission status as one master unit.

Next, this master unit outputs a parity bit, determines the master address of other unit, and starts outputting a slave address field.

If one unit occupies the bus as the master, the address specified by the IEBBnUAR register is output.

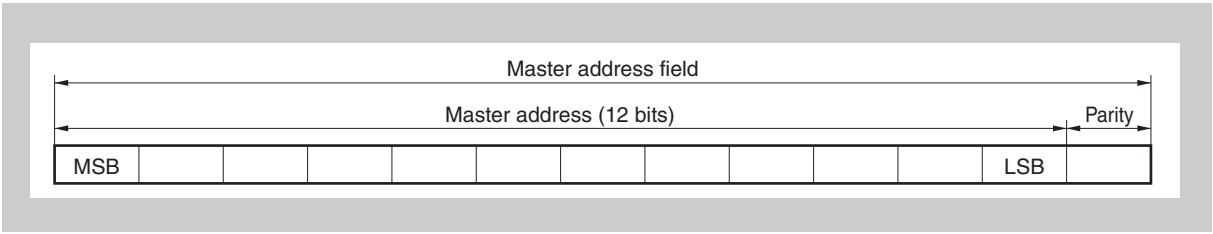


Figure 30-57 Master address field

(4) Slave address field

The master outputs the address of the unit with which it is to communicate.

The configuration of the slave address field is shown in *Figure 30-58*.

A parity bit is output after a 12-bit slave address has been transmitted to prevent the wrong slave address from being received by mistake. Next, the master unit detects an $\overline{ACK}$ signal from the slave unit to confirm that the slave unit exists on the bus. The master unit starts outputting the control field after detecting the $\overline{ACK}$ signal. During broadcast communication, however, the master does not confirm the acknowledge bit and instead starts outputting the control field.

The slave unit outputs the $\overline{ACK}$ signal if its slave address matches and if the slave detects that the parities of both the master address and slave address are even. The slave unit judges that the master address or slave address has not been correctly received and outputs the NACK signal if the parities are odd. At this time, the master unit is in the standby (monitor) status, and communication ends.

During broadcast communication, the slave address is used to identify group-unit broadcast or all-unit broadcast, as follows:

If the slave address is FFFH: All-unit broadcast communication

If the slave address is not FFFH: Group-unit broadcast communication

Note The group No. during group-unit broadcasting communication is the value of the higher 4 bits of the slave address.

If one unit occupies the bus as the master, the address specified by the IEBBnSAR register is output.

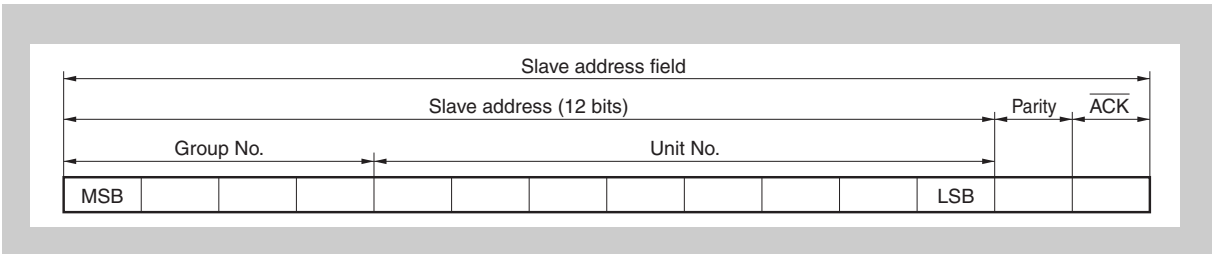


Figure 30-58 Slave address field

(5) Control data field

The master uses this field to output the operation it requires the slave to perform.

The configuration of the control field is shown in *Figure 30-59*.

If the parity following the control bit is even and the slave unit can execute the function required by the master unit, the slave unit outputs an $\overline{\text{ACK}}$ signal and starts outputting the message length field. If the slave unit cannot execute the function required by the master unit even if the parity is even, or if the parity is odd, the slave unit outputs the NACK signal, and returns to the standby (monitor) status.

The master unit starts outputting the message length field after detecting the $\overline{\text{ACK}}$ signal.

If the master detects the NACK signal, the master unit enters the standby status, and communication ends. During broadcast communication, however, the master unit does not confirm the acknowledge bit and starts outputting the message length field.

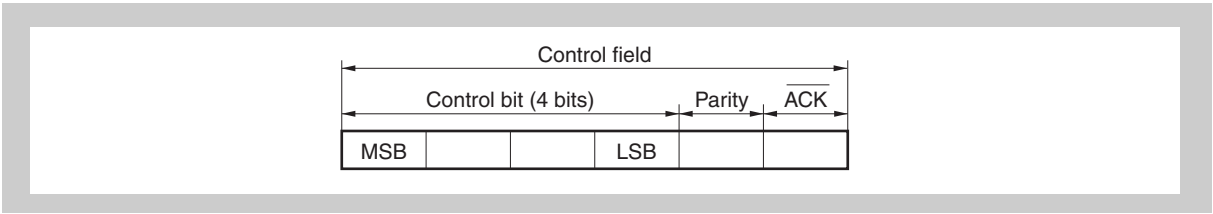


Figure 30-59 Control field

The contents of the control bits are shown below.

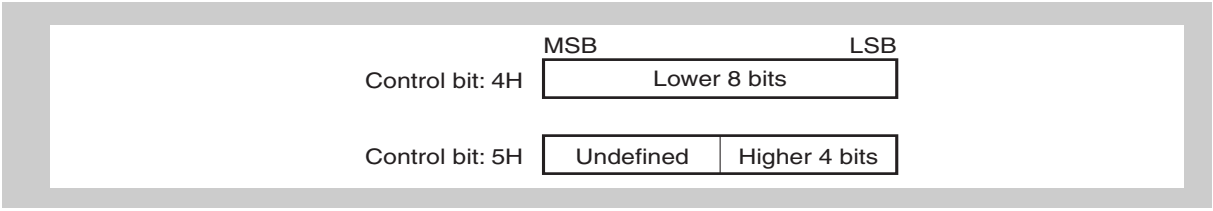
Table 30-59 Control bit contents

Bit 3[a]	Bit 2	Bit 1	Bit 0	Function
0	0	0	0	Read slave status
0	0	0	1	Undefined
0	0	1	0	Undefined
0	0	1	1	Read data and lock[b]
0	1	0	0	Read lock address (lower 8 bits)[c]
0	1	0	1	Lock address reading (higher 4 bits)[c]
0	1	1	0	Slave status reading and unlocking[b]
0	1	1	1	Read data
1	0	0	0	Undefined
1	0	0	1	Undefined
1	0	1	0	Command writing and locking[b]
1	0	1	1	Data writing and locking[b]
1	1	0	0	Undefined
1	1	0	1	Undefined
1	1	1	0	Write command
1	1	1	1	Write data

a) The message length bit of the message length field and data transfer direction of the data field change as follows depending on the value of bit 3 (MSB).
If bit 3 is 1: Transfer from master unit to slave unit
If bit 3 is 0: Transfer from slave unit to master unit

b) This is a control bit that specifies locking or unlocking. (For details, see *30.7.7 (4) "Locking and unlocking"*.)

c) The lock address is transferred in 1-byte (8-bit) units and is configured as follows:



If the control bit received from the master unit is not as shown in *Table 30-60*, the unit locked by the master unit rejects acknowledging the control bit, and outputs the NACK signal.

Table 30-60 Control field for locked slave unit

Bit 3	Bit 2	Bit 1	Bit 0	Function
0	0	0	0	Read slave status
0	1	0	0	Lock address reading (lower 8 bits)
0	1	0	1	Lock address reading (higher 4 bits)

In addition, units for which locking is not set up by the master unit reject acknowledgment and output a NACK signal when the control data shown in *Table 30-61* is acknowledged.

Table 30-61 Control field for unlocked slave unit

Bit 3	Bit 2	Bit 1	Bit 0	Function
0	1	0	0	Lock address reading (lower 8 bits)
0	1	0	1	Lock address reading (higher 4 bits)

If one unit occupies the bus as the master, the value set to the IEBBnTCD register is output.

Table 30-62 Control field $\overline{ACK}$ signal response conditions (when the received control data is H, 3H, 4H, 5H, 6H, or 7H)

Communication target (IEBBnUSR.IEBBnSRQF bit) Slave specification = 1 No specification = 0	Lock status (IEBBnUSR.IEBBnLCKF bit) Lock = 1 No lock = 0	Master unit judgment (IEBBnPAR register match) Lock request unit = 1 Other = 0	Slave transmission enabled (IEBBnBCR.IEBBnSTXE bit)	Slave reception enabled (IEBBnBCR.IEBBnSRXE bit)	Received control data					
					0H	3H	4H	5H	6H	7H
1	0	don't care	0	don't care	A	N	N	N	A	N
			1		A	A	N	N	A	A
	1	0	don't care		A	N	A	A	N	N
		1	0		A	N	A	A	A	N
			1		A	A	A	A	A	A
Other than the above					N					

Note A: Slave transmission is performed. (The $\overline{ACK}$ signal is returned.)

N: Slave transmission is not performed. (The NACK signal is returned.)

Caution If the received control data is other than the data shown in the above table, x is unconditionally assumed. (Slave transmission is not performed (and the NACK signal is returned).)

Table 30-63 Control field $\overline{ACK}$ signal response conditions (when the received control data is AH, BH, EH, or FH)

Communication target (IEBBnUSR.IEBBnSRQF bit) Slave specification = 1 No specification = 0	Lock status (IEBBnUSR.IEBBnLCKF bit) Lock = 1 No lock = 0	Master unit judgment (IEBBnPAR register match) Lock request unit = 1 Other = 0	Slave transmission enabled (IEBBnBCR.IEBBnSTXE bit)	Slave reception enabled (IEBBnBCR.IEBBnSRXE bit)	Received control data			
					AH	BH	EH	FH
1	0	don't care	don't care	1	A			
	1	1						
Other than the above					N			

Note A: Slave transmission is performed. (The $\overline{ACK}$ signal is returned.)

N: Slave transmission is not performed. (The NACK signal is returned.)

Caution If the received control data is other than the data shown in the above table, x is unconditionally assumed. (Slave transmission is not performed (and the NACK signal is returned).)

(6) Message length field

This field is output by the transmission side to inform the reception side of the number of bytes of the transmit data.

The configuration of the message length field is shown in *Figure 30-60*.

Table 30-64 shows the relationship between the message length bit and the number of transmission data bytes.

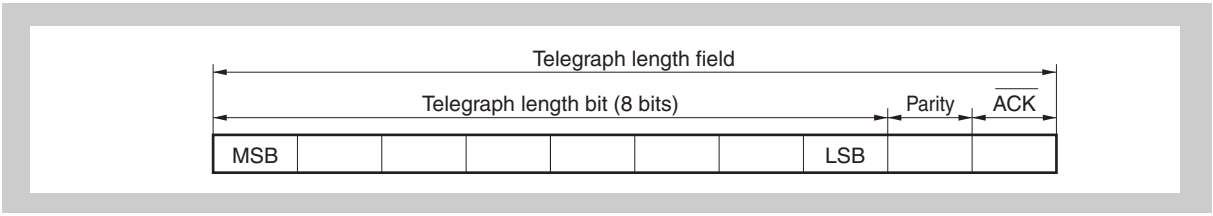


Figure 30-60 Message length field

Table 30-64 Contents of the message length bit

Message length bit (hexadecimal)	Number of transmission data bytes
01H	1 byte
02H	2 bytes
...	...
FFH	255 bytes
00H	256 bytes

The operation of the message length field differs depending on whether the master transmits data (when control bit 3 is 1) or receives data (when control bit 3 is 0).

(a) During master transmission

The message length bit and parity bit are output by the master unit and the synchronization signals of bits are output by the master unit. When the slave unit detects that the parity is even, it outputs the $\overline{\text{ACK}}$ signal, and starts outputting the data field. During broadcast communication, however, the slave unit outputs the NACK signal.

If the parity is odd, the slave unit judges that the message length bit has not been correctly received, outputs the NACK signal, and returns to the standby (monitor) status. At this time, the master unit also returns to the standby status, and communication ends.

(b) Master reception

The message length bit and parity bit are output by the slave unit and the synchronization signals of bits are output by the master unit. If the master unit detects that the parity bit is even, it outputs the $\overline{\text{ACK}}$ signal.

If the parity bit is odd, the master unit judges that the message length bit has not been correctly received, outputs the NACK signal, and returns to the standby status. At this time, the slave unit also returns to the standby status, and communication ends.

(7) Data field

This is data output by the transmission side.

The master unit transmits or receives data to or from a slave unit by using the data field.

The configuration of the data field is shown below.

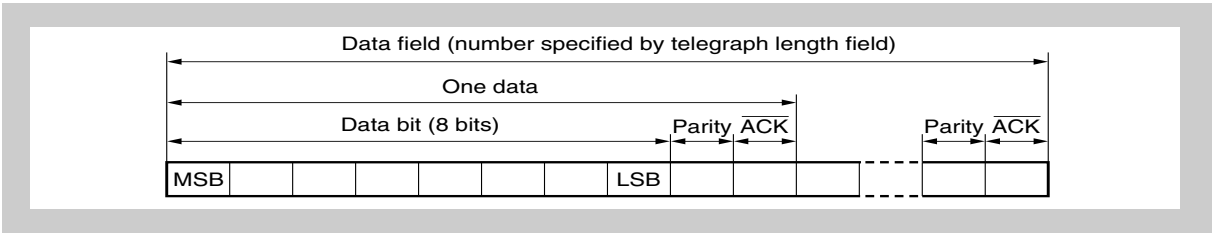


Figure 30-61 Data field

Following the data bit, the parity bit and acknowledge bit are output by the master unit and slave unit, respectively.

Use broadcast communication only when the master unit transmits data. At this time, the acknowledge bit is ignored.

The operation differs as follows depending on whether the master transmits or receives data.

(a) During master transmission

When the master units writes data to a slave unit, the master unit transmits the data bit and parity bit to the slave unit. If the parity is even and the received data is not stored in the IEBBnDR register when the slave unit has received the data bit and parity bit, the slave unit outputs an $\overline{\text{ACK}}$ signal. If the parity is odd or the received data is stored in the IEBBnDR register, the slave unit rejects receiving the data, and outputs the NACK signal.

If the slave unit outputs the NACK signal, the master unit transmits the same data again. This operation continues until the master detects the $\overline{\text{ACK}}$ signal from the slave unit, or the data exceeds the maximum number of transmit bytes.

If there is more data and the maximum number of transmission bytes is not exceeded when the parity is even and when the slave unit outputs the $\overline{\text{ACK}}$ signal, the master unit transmits the next data.

During broadcast communication, the slave unit outputs the NACK signal, and the master unit transfers 1 byte of data at a time. If the parity is odd or the IEBBnDR register is storing received data after the slave unit receives the data bit and parity bit during broadcast communication, the slave unit judges that reception has not been performed correctly, and stops reception.

(b) Master reception

When the master unit reads data from a slave unit, the master unit outputs a sync signal corresponding to all the read bits.

The slave unit outputs the contents of the data and parity bits to the bus in response to the sync signal from the master unit.

The master unit reads the data and parity bits output by the slave unit, and checks the parity.

If the parity is odd or the IEBBnDR register is storing received data, the master unit rejects accepting the data, and outputs the NACK signal. If the maximum number of transmission bytes is within the value that can be transmitted in one communication frame, the master unit rereads the same data.

If the parity is even and the IEBBnDR register is not storing received data, the master unit accepts the data and outputs the $\overline{\text{ACK}}$ signal. If the maximum number of transmission bytes is within the value that can be transmitted in one frame, the master unit reads the next data.

Caution During broadcast communication, do not perform master reception. If you do this, the slave unit cannot be defined and data transfers cannot be performed correctly.

Note that, due to the IEBBn specifications, overrun errors can occur. Therefore, even if reading the IEBBnDR register is late during individual communication and the system has reached the timing for receiving the next data (the overrun status), data can be retransmitted from the master unit by returning a NACK signal, which makes it possible to buy time for reading the IEBBnDR register. However, during broadcast communication, because no $\overline{\text{ACK}}$ signal is output from the slave unit and the master unit ignores $\overline{\text{ACK}}$ signals, even if reading the IEBBnDR register is late, no data is retransmitted from the master. Therefore, for IEBBn, if an overrun occurs during broadcast communication, normal reception is not possible, an overrun error occurs, and an interrupt request (for a communication error) is output.

(8) Parity bit

The parity bit is used to make sure that the transmission data has no error.

The parity bit is appended to each data of the master address, slave address, control, message length, and data bits.

The parity is an even parity. If the number of data bits that are ‘1’ is odd, the parity bit is ‘1’. If the number of data bits that are ‘1’ is even, the parity bit is ‘0’.

(9) Acknowledge bit

During normal communication (communication from one unit to another), an acknowledge bit is appended to the following locations to check whether the data has been correctly received.

- End of slave address field
- End of control field
- End of message length field
- End of data field

The definition of the acknowledge bit is as follows.

0: The transmission data is recognized. ($\overline{\text{ACK}}$ signal)

1: The transmission data is not recognized. (NACK signal)

During broadcast communication, however, the contents of the acknowledge bit are ignored.

(a) Last acknowledge bit of the slave address field

The last acknowledge bit of the slave address field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the master address bit or slave address bit is incorrect
- If a timing error (an error in the bit format) occurs
- If a slave unit does not exist

(b) Last acknowledge bit of the control field

The last acknowledge bit of the control field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the control bit is incorrect
- If control bit 3 is 1 (write operation) when the slave reception enable flag (the IEBBnBCR.IEBBnSRXE bit) is not set (to 1)

  (For details, see *30.3.2 (1) "IEBBnBCR - IEBBn bus control register"*.)
- If control bit data is read (3H, 7H) when the slave reception enable flag (the IEBBnBCR.IEBBnSRXE bit) is not set (to 1)

  (For details, see *30.3.2 (1) "IEBBnBCR - IEBBn bus control register"*.)
- If a unit other than one that has set locking requests 3H, 6H, 7H, AH, BH, EH, or FH of the control bit when locking is set
- If the control bit indicates reading of lock addresses (4H, 5H) even when locking is not set
- If a timing error occurs
- If the control bit is undefined

Cautions

1. The $\overline{\text{ACK}}$ signal is always returned when the control data of the slave status request is received, if the IEBBnSTXE bit = 0.
2. The NACK signal is returned by the acknowledge bit in the control field when the control data for data/command writing is received, even if the IEBBnSRXE bit = 0.

   Slave reception can be disabled (communication stopped) by the IEBBnSRXE bit only in the case of individual communication. In the case of broadcast communication, communication is maintained and the data interrupt (IEBBTD) or completion interrupt (IEBBTSTA) is generated.

(c) Last acknowledge bit of message length field

The last acknowledge bit of the message length field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the message length bit is incorrect
- If a timing error occurs

(d) Last acknowledge bit of the data field

The last acknowledge bit of the data field serves as a NACK signal in any of the following cases, and transmission is stopped.

- If the parity of the data bit is incorrect[a]
- If a timing error occurs after the preceding acknowledge bit has been transmitted
- If the received data is stored in the IEBBnDR register and no more data can be received[a]

a) In this case, when the communication executed is individual communication, if the maximum number of transmit bytes is within the value that can be transmitted in one frame, the transmission side executes transmission of that data field again. For broadcast communication, the transmission side does not execute transmission again, a communication error occurs on the reception side and reception stops.

30.7.7 Transfer data

(1) Slave status

The master unit can learn why the slave unit did not return the $\overline{\text{ACK}}$ signal by reading the slave status.

The slave status is determined according to the result of the last communication the slave unit has executed.

All the slave units can supply information on the slave status.

The configuration of the slave status is shown below.

MSB LSB

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Bit 0[a]	Function
0	Transmit data is not written in IEBBnDR register
1	Transmit data is written in IEBBnDR register

Bit 1	Function
0	Receive data is not stored in IEBBnDR register
1	Receive data is stored in IEBBnDR register

Bit 2	Function
0	Unit is not locked
1	Unit is locked

Bit 3	Function
0	Fixed to 0

Bit 4[b]	Function
0	Slave transmission is stopped
1	Slave transmission is ready

Bit 5	Function
0	Fixed to 0

Bit 7	Bit 6	Function	
0	0	Mode 0	Indicates the highest mode supported by the unit[c].
0	1	Mode 1	
1	0	Mode 2	
1	1	Not used	

a) After reset: Bit 0 is set to 1.

b) When the V850E2/Sx4-H serves as a slave unit, this bit corresponds to the status indicated by IEBBnBCR.IEBBnSTXE bit.

c) Bits 7 and 6 are fixed to “10” because the V850E2/Sx4-H can support modes 1 and 2.

Figure 30-62 Slave status bit configuration

(2) Lock address

When the lock address is read (control bit: 4H or 5H), the address (12 bits) of the master unit that has issued the lock instruction is configured in 1-byte units as shown below and read.

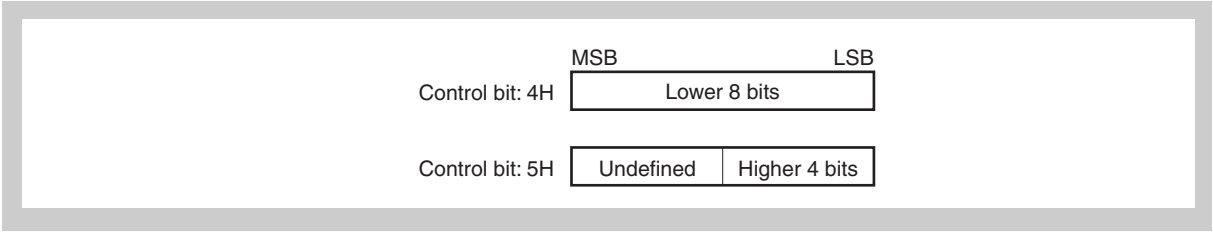


Figure 30-63 Lock address configuration

(3) Data

If the control bit indicates reading of data (3H or 7H), the data in the data buffer of the slave unit is read by the master unit.

If the control bit indicates writing of data (BH or FH), the data received by the slave unit is processed according to the operation rule of that slave unit.

(4) Locking and unlocking

The lock function is used when a message is transferred in two or more communication frames.

The unit that is locked does not receive data from units other than the one that has locked the unit (does not receive broadcast communication).

A unit is locked or unlocked as follows.

(a) Lock setting

If the communication frame is completed without succeeding to transmit or receive data of the number of bytes specified by the message length bit after the message length field has been transmitted or received ($\overline{ACK}$ = 0) by the control bit that specifies locking (3H, AH, or BH), the slave unit is locked by the master unit. At this time, the bit (bit 2) in the byte indicating the slave status is set to ‘1’.

(b) Unlocked

After transmitting or receiving data of the number of data bytes specified by the message length bit in one communication frame by the control bit that has specified locking (3H, AH, or BH), or the control bit that has specified unlocking (6H), the slave unit is unlocked by the master unit. At this time, the bit related to locking (bit 2) in the byte indicating the slave status is reset to ‘0’.

Locking or unlocking is not performed during broadcast communication.

Locking and unlocking conditions are shown below.

Table 30-65 Setting conditions:

Control data	Broadcast communication		Individual communication	
	End of communication	End of frame	End of communication	End of frame
3H, 6H[a]	–	–	Cannot be locked	Lock set
AH, BH	Cannot be locked	Cannot be locked	Cannot be locked	Lock set
0H, 4H, 5H, EH, FH	Cannot be locked	Cannot be locked	Cannot be locked	Cannot be locked

a) The frame end of control data 6H (slave status read/unlock) occurs when the parity in the data field is odd, and when the NACK signal from the IEBus unit is repeated with up to the maximum number of transfer bytes being output.

Table 30-66 Unlocking conditions (while locked)

Control data	Broadcast communication fromthe lock request unit		Individual communication fromthe lock request unit	
	End of communication	End of frame	End of communication	End of frame
3H, 6H[a]	–	–	Unlocked	Remains locked
AH, BH	Unlocked	Unlocked	Unlocked	Remains locked
0H, 4H, 5H, EH, FH	Remains locked	Remains locked	Remains locked	Remains locked

a) The frame end of control data 6H (slave status read/unlock) occurs when the parity in the data field is odd, and when the NACK signal from the IEBus unit is repeated with up to the maximum number of transfer bytes being output.

30.7.8 Bit format

The format of the bits constituting the communication frame of the IEBus is shown below.

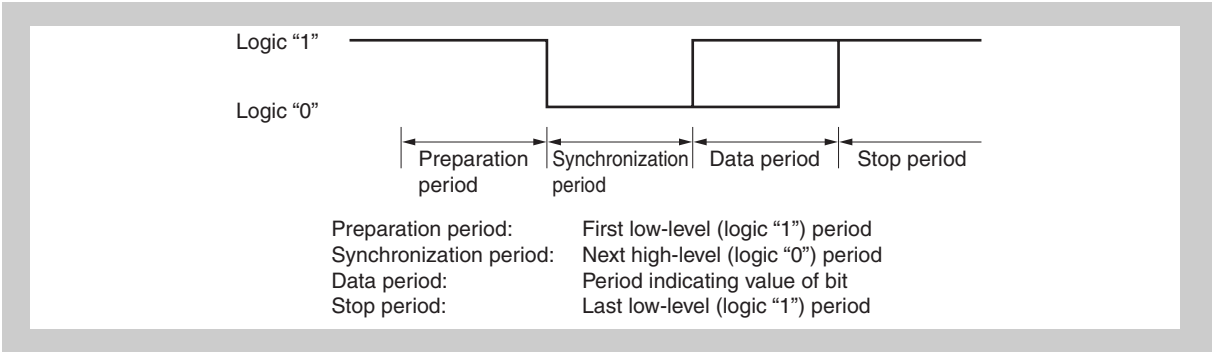


Figure 30-64 IEBus bit format

The synchronization period and data period are almost equal to each other in length.

The IEBus synchronizes each bit. The specifications on the time of the entire bit and the time related to the period allocated to that bit differ depending on the type of transmit bit, or whether the unit is the master unit or a slave unit. The master and slave units monitor whether each period (preparation period, synchronization period, data period, and stop period) is output for the specified time while they are in communication. If a period is not output for the specified time, the master and slave units report a timing error, immediately terminate communication, and enter the standby status.

Chapter 31 Key Return Function (KR)

This chapter describes the key return function (KR).

The first section describes the properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names.
The subsequent sections describe the features that apply to all implementations.

31.1 V850E2/Sx4-H KR Features

Instances

This microcontroller has following number of instances of KR:

Table 31-1 Instances of KR

KR	
Number of instances	1
Name	KR0

Instances index n

Throughout this chapter, the individual instances of KR are identified by the index “n” (n = 0), for example, KRnM for the key return mode register.

Register address

The KRn register addresses are given as addresses offset from the base address <KRn_base>.
The base address <KRn_base> of KRn is listed in the following table:

Table 31-2 Register base address <KRn_base>

KRn	<KRn_base> address
KR0	$FF82\ B000_H$

Clock supply

The following clock is supplied to KR:

Table 31-3 KRn clock supply

KRn	Clock	Connected to:
KR0	PCLK	Clock generator CKSCLK_A02

Interrupts and DMA

KR can generate the following interrupt requests and DMA requests:

Table 31-4 KRn interrupt request and DMA request

KRn signals	Function	Connected to:
KR0:		
KRnTIKR	Key interrupt	Interrupt controller INTKR0 DMA controller trigger 115

KR hardware reset

KR and its registers are initialized by the following reset signal:

Table 31-5 KRn reset signal

KRn	Reset signal
KR0	System reset SYSRES

I/O signals

The I/O signals of KR are listed in the following table:

Table 31-6 KRn I/O signals

KR signal	Function	Connected to:
KR0TPKR7 to KR0TPKR0	Key input signals	Ports KR0I0 to KR0I7[a]

a) The key input signals are input via noise filters. For details about noise filters, see *2.5 “Port Filters”*.

31.2 Functional Overview

Features summary The key return function has the following feature:

A key interrupt request signal (KRnTIKR) can be generated by inputting a low level to any of the eight key input pins (KRnTPKR0 to KRnTPKR7).

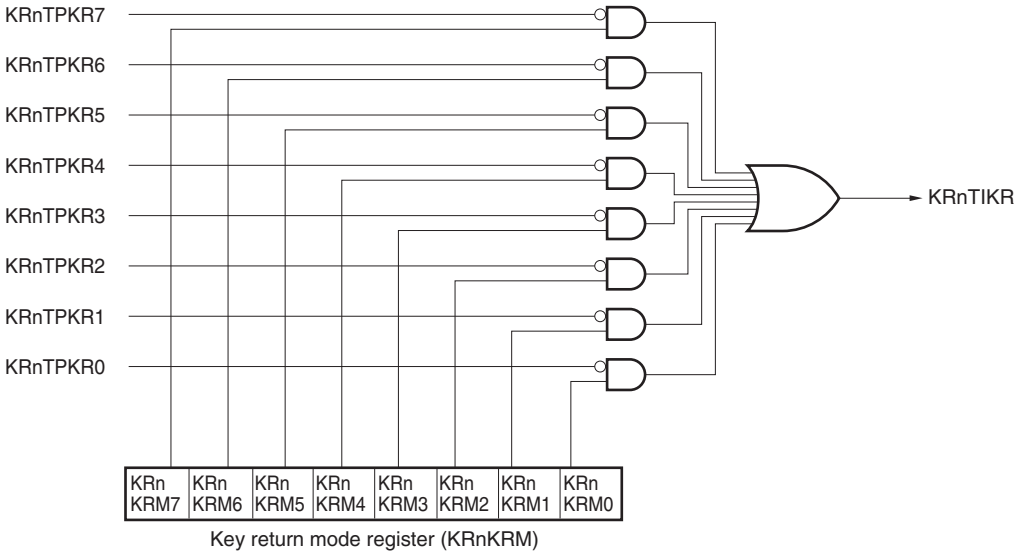


Figure 31-1 Block diagram of the key return function

31.3 Functional Description

31.3.1 Interrupt request KRnTIKR

The interrupt request KRnTIKR is generated when a low level is input to key input pin KRnTPKR[7:0] while input to pin KRnTPKR[7:0] is enabled (KRnM.KRnKRM[7:0] = 1).

The following figure shows the interrupt request generation:

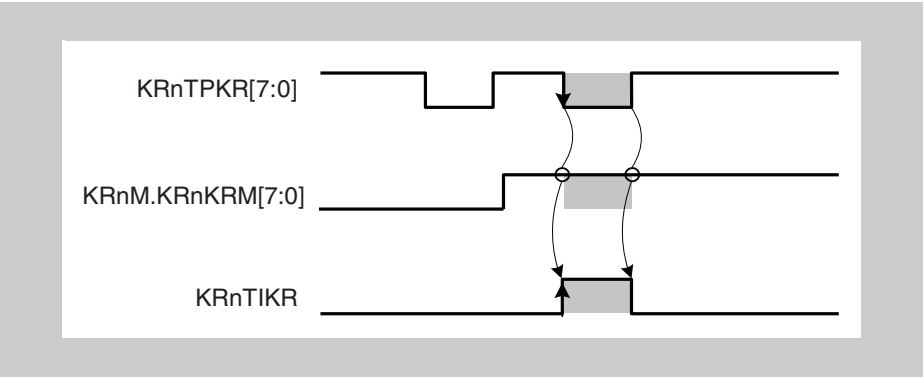


Figure 31-2 Interrupt request generation

Cautions

1. The interrupt request KRnTIKR is not generated if another key input changes from high to low while the low level is input to a key input pin KRnTPKR[7:0].
2. An unintended interrupt request might be generated if the key input value changes while the KRnKRM[7:0] setting is changed.

   Therefore, mask (disable) KRnTIKR of the interrupt controller before changing KRnM.KRnKRM[7:0].

31.4 Registers

This section provides a description of the key return function register.

31.4.1 Key return function register overview

The key return function is controlled by the following register:

Table 31-7 Key return function register overview

Register name	Symbol	Address
Key return mode register	KRnM	<KRn_base>

<KRn_base> <KRn_base> of the KRn are defined in the first section of this chapter under the key word “Register addresses”.

31.4.2 Key return function register details

(1) KRnM - Key return mode register

This register enables/disables the key input signal detection.

Access This register can be read or written in 8-bit or 1-bit units.

Address <KRn_base>

Initial Value Reset input initializes this register to 0000 0000_H.

7	6	5	4	3	2	1	0
KRn KRM7	KRn KRM6	KRn KRM5	KRn KRM4	KRn KRM3	KRn KRM2	KRn KRM1	KRn KRM0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-8 KRnM register contents

Bit position	Bit name	Function
7 to 0	KRn KRMm	Enables/disables the key input signal detection 0: Disable 1: Enable

Chapter 32 A/D Converter (ADCA)

This chapter describes A/D converter A (ADCA).

The first section describes the properties specific to the V850E2/Sx4-H, such as instances, register base addresses, and input/output signal names. The subsequent sections describe the features that apply to all implementation.

Caution The V850E2/Sx4-H does not support the following features:

- 12-bit resolution
  Be sure to set the ADCAnCTL1.ADCAnCTYP bit to 1.
- Channel S/H function
  Accessing the ADCAnSHCTL register is prohibited.

32.1 V850E2/Sx4-H ADCA Features

Instances This microcontroller has the following number of instances of ADCA:

Table 32-1 Instances of ADCA

	ADCA
Number of instances	1
Name	ADCA0

Instances index n Throughout this chapter, the individual instances of ADCA is identified by the index "n" (n = 0), for example, ADCAnCTL0 for A/D converter mode control register 0.

Channel group index i ADCA has three A/D conversion channel groups, abbreviated with CG. Throughout this chapter, the individual channel group is identified by the index "i" (i = 0 to 2), for example, ADCAnIOCi for the CGi interrupt controller register.

Channel index m Each ADCA has several A/D conversion channels. Throughout this chapter, the individual channels of each ADCA are identified by the index “m”, for example, ADCAnCmCR for the A/D converter conversion result register m.

The number of channels of the ADCA instances for each device are given in the following table:

Table 32-2 ADCAn channel indices

<R>

ADCAn	Channel index		
	V850E2/FG4-H	V850E2/FJ4-H	V850E2/FK4-H
ADCA0	m = 0 to 7	m = 0 to 15	m = 0 to 15

Register addresses

All ADCAn register addresses are given as addresses offset from the individual base address <ADCAn_base>.
The base address <ADCAn_base> of each ADCAn is listed in the following table:

Table 32-3 Register base addresses <ADCAn_base>

ADCAn	<ADCAn_base> address
ADCA0	FF81 D000$_H$

Clock supply

The following clocks are supplied to ADCA:

Table 32-4 ADCAn clock supply

ADCAn	Clock	Connected to:
ADCA0	PCLK	Clock generator CKSCLK_012

Interrupts and DMA

ADCA can generate the following interrupt requests and DMA requests:

Table 32-5 ADCAn interrupt requests

ADCAn signals	Function	Connected to:
ADCA0:		
ADCATINT0	End of conversion on CG0	Interrupt controller INTADCA0I0 DMA controller trigger 38
ADCATINT1	End of conversion on CG1	Interrupt controller INTADCA0I1 DMA controller trigger 39
ADCATINT2	End of conversion on CG2	Interrupt controller INTADCA0I2 DMA controller trigger 40
ADCATLLT	Last conversion	Interrupt controller INTADCA0LLT DMA controller trigger 41
ADCATERR	Error interrupt	Interrupt controller INTADCA0ERR

ADCA hardware reset

ADCA and its registers are initialized by the following reset signal:

Table 32-6 ADCAn reset signal

ADCAn	Reset signal
ADCAn	System reset SYSRES

I/O signals The I/O signals of ADCA are listed in the following table:

Table 32-7 ADCAn I/O signals

ADCAn signal	Function	Connected to:
ADCA0:		
ANIm	Analog inputs	Port ADCA0Im
ADCATTRGi	Hardware trigger for CGi	For details about hardware trigger expansion i, see *32.1.1 "Hardware trigger expansion" on page 2331*.
<R> AVREFP	Positive analog reference voltage	Port A0VREFP
<R> AVREFM	Negaive analog reference voltage	Port A0VREFM

32.1.1 Hardware trigger expansion

All A/D converter hardware trigger inputs ADCATTRGi are equipped with a hardware trigger expansion module that enables up to 16 signals ADCAnTTINi[15:00] to trigger an A/D conversion process.

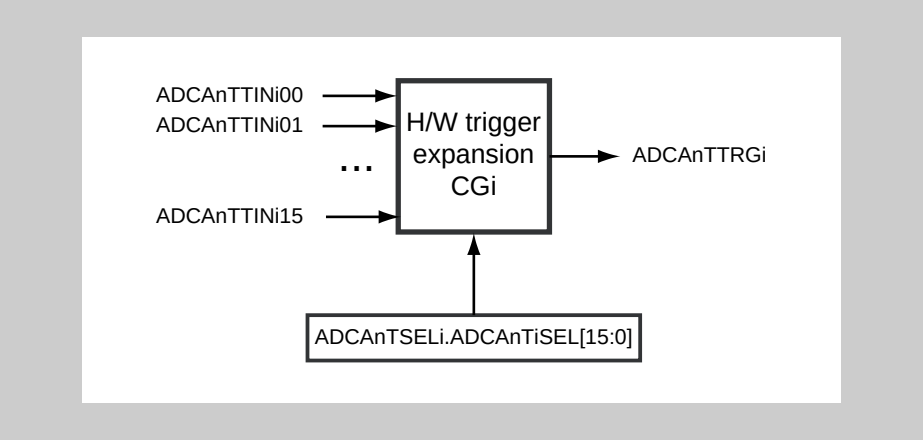


Figure 32-1 Hardware trigger expansion module

(1) ADCAn hardware trigger selection

Active hardware triggers can be selected by using the ADCAnTSELi.TiSEL[15:0] bits.

Caution It is only allowed to select a single hardware trigger for each ADCAn hardware trigger input ADCATTRGi.
Thus do not set more than one bit of each ADCAnTSELi register to 1 at the same time.

(2) ADCAn hardware trigger edge setting

To enable the rising edge being input to ADCAnTTRGi as a hardware trigger, the ADCAnCTL1.ADCAnTiETS[1:0] bits must be set.

(3) ADCAn hardware triggers

This product has various hardware trigger functions. The table below lists the connection destination of the hardware trigger ADCA0TRGi for each channel group (CGi).

Table 32-8 ADCA0 H/W trigger signals (1/2)

ADCA0 channel group	Trigger input signals			ADCA0 trigger signal
	Name	Control bit	Connected to:	
CG0	ADCA0TTIN000	ADCA0TSEL0.ADCA0T0SEL[00]	Port ADCA0TRG0	ADCA0TTRG0
	ADCA0TTIN001	ADCA0TSEL0.ADCA0T0SEL[01]	Port INTP2	
	ADCA0TTIN002	ADCA0TSEL0.ADCA0T0SEL[02]	Port INTP5	
	ADCA0TTIN003	ADCA0TSEL0.ADCA0T0SEL[03]	Not connected	
	ADCA0TTIN004	ADCA0TSEL0.ADCA0T0SEL[04]	Not connected	
	ADCA0TTIN005	ADCA0TSEL0.ADCA0T0SEL[05]	INTTAUB2I15	
	ADCA0TTIN006	ADCA0TSEL0.ADCA0T0SEL[06]	Not connected	
	ADCA0TTIN007	ADCA0TSEL0.ADCA0T0SEL[07]	Not connected	
	ADCA0TTIN008	ADCA0TSEL0.ADCA0T0SEL[08]	INTTAUJ0I3	
	ADCA0TTIN009	ADCA0TSEL0.ADCA0T0SEL[09]	Not connected	
	ADCA0TTIN010	ADCA0TSEL0.ADCA0T0SEL[10]	Not connected	
	ADCA0TTIN011	ADCA0TSEL0.ADCA0T0SEL[11]	Not connected	
	ADCA0TTIN012	ADCA0TSEL0.ADCA0T0SEL[12]	Not connected	
	ADCA0TTIN013	ADCA0TSEL0.ADCA0T0SEL[13]	Not connected	
	ADCA0TTIN014	ADCA0TSEL0.ADCA0T0SEL[14]	Not connected	
	ADCA0TTIN015	ADCA0TSEL0.ADCA0T0SEL[15]	Not connected	
CG1	ADCA0TTIN100	ADCA0TSEL1.ADCA0TISEL[00]	Port ADCA0TRG1	ADCA0TTRG1
	ADCA0TTIN101	ADCA0TSEL1.ADCA0TISEL[01]	Port INTP1	
	ADCA0TTIN102	ADCA0TSEL1.ADCA0TISEL[02]	Port INTP4	
	ADCA0TTIN103	ADCA0TSEL1.ADCA0TISEL[03]	Not connected	
	ADCA0TTIN104	ADCA0TSEL1.ADCA0T1SEL[04]	Not connected	
	ADCA0TTIN105	ADCA0TSEL1.ADCA0T1SEL[05]	INTTAUB2I15	
	ADCA0TTIN106	ADCA0TSEL1.ADCA0T1SEL[06]	Not connected	
	ADCA0TTIN107	ADCA0TSEL1.ADCA0T1SEL[07]	Not connected	
	ADCA0TTIN108	ADCA0TSEL1.ADCA0T1SEL[08]	INTTAUJ0I3	
	ADCA0TTIN109	ADCA0TSEL1.ADCA0T1SEL[09]	Not connected	
	ADCA0TTIN110	ADCA0TSEL1.ADCA0T1SEL[10]	Not connected	
	ADCA0TTIN111	ADCA0TSEL1.ADCA0T1SEL[11]	Not connected	
	ADCA0TTIN112	ADCA0TSEL1.ADCA0T1SEL[12]	Not connected	
	ADCA0TTIN113	ADCA0TSEL1.ADCA0T1SEL[13]	Not connected	
	ADCA0TTIN114	ADCA0TSEL1.ADCA0T1SEL[14]	Not connected	
	ADCA0TTIN115	ADCA0TSEL1.ADCA0T1SEL[15]	Not connected	

Table 32-8 ADCA0 H/W trigger signals (2/2)

ADCA0 channel group	Trigger input signals			ADCA0 trigger signal
	Name	Control bit	Connected to:	
CG2	ADCA0TTIN200	ADCA0TSEL2.ADCA0T2SEL[00]	Port ADCA0TRG2	ADCA0TTRG2
	ADCA0TTIN201	ADCA0TSEL2.ADCA0T2SEL[01]	Port INTP0	
	ADCA0TTIN202	ADCA0TSEL2.ADCA0T2SEL[02]	Port INTP3	
	ADCA0TTIN203	ADCA0TSEL2.ADCA0T2SEL[03]	Not connected	
	ADCA0TTIN204	ADCA0TSEL2.ADCA0T2SEL[04]	Not connected	
	ADCA0TTIN205	ADCA0TSEL2.ADCA0T2SEL[05]	INTTAUB2I15	
	ADCA0TTIN206	ADCA0TSEL2.ADCA0T2SEL[06]	Not connected	
	ADCA0TTIN207	ADCA0TSEL2.ADCA0T2SEL[07]	Not connected	
	ADCA0TTIN208	ADCA0TSEL2.ADCA0T2SEL[08]	INTTAUJ0I3	
	ADCA0TTIN209	ADCA0TSEL2.ADCA0T2SEL[09]	Not connected	
	ADCA0TTIN210	ADCA0TSEL2.ADCA0T2SEL[10]	Not connected	
	ADCA0TTIN211	ADCA0TSEL2.ADCA0T2SEL[11]	Not connected	
	ADCA0TTIN212	ADCA0TSEL2.ADCA0T2SEL[12]	Not connected	
	ADCA0TTIN213	ADCA0TSEL2.ADCA0T2SEL[13]	Not connected	
	ADCA0TTIN214	ADCA0TSEL2.ADCA0T2SEL[14]	Not connected	
	ADCA0TTIN215	ADCA0TSEL2.ADCA0T2SEL[15]	Not connected	

32.1.2 Operation in power save mode

- HALT mode

  The A/D conversion operation is performed continuously in HALT mode. When HALT mode is released by maskable interrupt input, the ADCAnCTL0 register and the ADCAnCmCR register retain their values.

32.2 Functional Overview

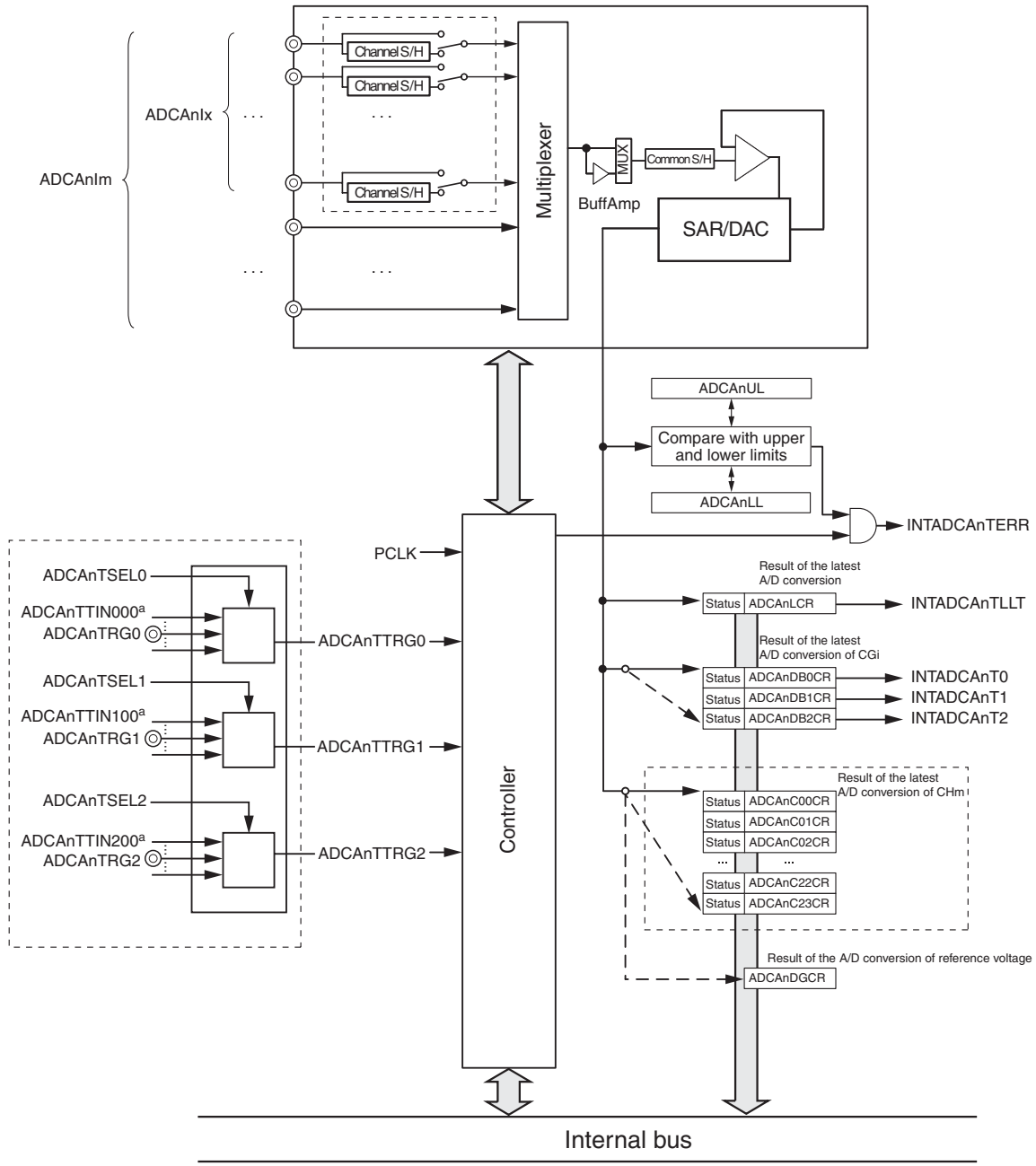
The A/D Converter A (ADCAn) converts analog input signals into digital values.

Features summary

The ADCAn has the following features:

- Support of 10-bit and 12-bit resolution
- A/D conversion based on successive approximation method
- Up to 24 analog input signals (depends on the product)
- Up to 3 channel groups with different priority levels
- One-shot conversion mode and continuous conversion mode (channel group 0 only)
- Auto repeat function (repeats the one-shot conversion mode 1 to 4 times)
- Software and hardware start trigger modes
- The hardware trigger source can be selected from multiple input signals
- The channel to which the A/D conversion end interrupt is generated at the end of A/D conversion can be set
- Three types of result check functions
- On-off switch function for buffer amplifier
- Discharge function for discharging the capacitors prior to executing conversion with a new sampling value

The following figure shows the main components of the ADCAn.



a) The connection destinations differ according to the product. See the first section of this chapter for details.

<R> **Figure 32-2 Block diagram of the ADCAn**

Note The parts enclosed in dotted lines are product dependent functions.

32.3 Functional Description

The A/D converter A (ADCAn) converts up to 24 analog input signals into digital values and supports 10-bit and 12-bit resolution (product dependent).

Note The 10-bit/12-bit resolution setting is applied in common to all the channels. The resolution cannot be changed for individual channels.

Channels and channel groups

Each of the input channels can be assigned to one of three channel groups (CG). The list of the input channels assigned to each CG is called a scan list (including diagnostic A/D conversion of CG0). The scan lists can be set easily with one register. The scan lists can also be set again during operation. All the A/D conversions for a scan list are called scan list conversion.

The ADCAn supports up to 3 differently-prioritized channel groups and 2 conversion modes:

- One-shot conversion mode: Executes scan list conversion only once. In one-shot conversion mode, scan list conversion is repeated the specified number of times (1 to 4 times).
- Continuous conversion mode: Repeats scan list conversion.

A/D conversion

The A/D conversion can be started using either software or hardware as the start trigger.

A multiplexer selects the channel to be converted and the common sample & hold circuit holds the voltage input.

The successive approximation register (SAR) stores the D/A converter output voltage to be compared with the analog input voltage as a 10-bit or 12-bit digital value.

After each successful conversion the INTADCAnTLLT signal is generated.

A/D conversion result registers

When the A/D conversion ends, the contents of the SAR register are stored in three registers, and the results of the latest A/D conversion, the latest A/D conversion of CGi, and the latest A/D conversion of channel m can be read separately.

Depending on the configuration, the ADCAn generates a conversion end interrupt after the A/D conversion of certain channels and/or at the end of the A/D conversion of all channels of a channel group.

Optional result check (product dependent)

The results of A/D conversions can be checked with the following functions in ADCAn.

- Conversion result overwrite check function
- Conversion result read flag function
- Conversion result upper/lower limit compare function

Optional discharge function (product dependent)

If required, the internal capacitor of the common sample & hold circuit can be discharged prior to every conversion.

On-off switch function for buffer amplifier

In order to reduce the load of the external analogue signal source, the signal can be connected to an internal buffer amplifier.

Charging the internal sampling capacitor during the A/D sampling period is accelerated by the buffer amplifier.

Self-diagnosis functions

Four (product dependent) self-diagnostic functions are provided to check that the ADCA works correctly and to detect analog input pins that are disconnected.

- A/D conversion circuit diagnosis
- Channel multiplexer diagnosis
- Open pin diagnosis
- Channel S/H diagnosis (product-dependent)

Configurable stabilization time

By setting an arbitrary value to the stabilization counter, the optimum stabilization time can be secured after the power is switched on.

32.3.1 Basic Operation

This section describes the basic procedure for an analog to digital conversion. More detailed descriptions are given in the following chapters.

1. To optimize the startup time after power on and after standby mode is released, adjust the stabilization time by specifying the stabilization counter ADCAnCNT register.
2. Before you enable the A/D converter (ADCAnCTL0.ADCAnCE = 1 is required), configure the power-on, resolution, ADCAn clock, trigger mode, conversion mode, interrupt generation, channel groups, etc. in the following registers:
   - ADCAnCTL1
   - ADCAnCGi
   - ADCAnIOCi
   - ADCAnTSELi (product dependent)
3. If you want to check that the A/D conversion results are within a certain value range, enable the conversion result upper/lower limit compare function for the desired channels (ADCAnCTL2.ADCAnRCKm) and specify the lower and upper limits for the ADCAnLL and ADCAnUL registers.
4. Discharge the capacitor of the common S/H circuit by setting ADCAnCTL1.ADCAnDISC to enable the discharge function.
5. Enable or disable the buffer amplifier by setting ADCAnCTL1.ADCAnBPC.
6. Enable the A/D converter by setting ADCAnCTL0.ADCAnCE to 1.

   The A/D converter is ready for A/D conversion after the stabilization time has elapsed after power on or standby mode release.
7. Depending on the configured trigger mode, A/D conversion is started by a channel group related start trigger:
   - by a software trigger (ADCAnTRGi.ADCAnSTTi = 1), or
   - by a hardware trigger (input signal ADCATTRGi)

   If the A/D conversion starts for multiple CGs, the order of A/D conversion depends on the priority of the CGs.
8. The A/D conversion end interrupt INTADCAnTi is generated when the conversion of the channel set in the ADCAnIOCi register ends.
9. Read the results from the A/D conversion result registers: ADCAnLCR, ADCAnDBiCR, and ADCAnCmCR.

10. Monitor the following registers:
    - ADCAnSTR1: To check whether the A/D conversion results are overwritten before being read according to the intended application.
    - ADCAnSTR0: To check whether the A/D conversion results are within the setting range (only if the conversion result upper/lower limit compare function is enabled).
11. Disable the A/D converter if you want to reconfigure it. To do so, set ADCAnCTL0.ADCAnCE to 0.

Note The self-diagnostic functions are described in *32.3.13 "Self-diagnosis functions" on page 2355*.

32.3.2 Clock usage

The ADCAn clock ADCAnTCLK is derived from PCLK. The division factor is specified in ADCAnCTL1.ADCAnFR[3:0].

32.3.3 Channels and channel groups

The input channels are configured as channel groups (CG). A scan list can be created for each CG through register settings, and also can be set again during operation. The conversion settings for a CG are applied to all the channels within that group.

ADCAn supports up to three channel groups (CGi, where i = 0 to 2). The channels of each CGi are specified with the ADCAnCGi register.

Note The conversion of a single channel can be performed by assigning only one input channel to a CG.

(1) Order of A/D conversion

Upon occurrence of the start trigger for a CG, the channels set to its scan list are converted in ascending order, starting from CH00 to CH23 (product dependent).

If A/D conversion requests for multiple CGs are pending, the CGs are converted in the following hierarchical order:
CG2 (highest priority) > CG1 > CG0 (lowest priority)

If the start trigger for a CG with a higher priority, or the trigger for ADCHALT mode is set, the current A/D conversion is halted. One of the following two methods can be selected for halting A/D conversion, according to the setting of ADCAnCTL1.ADCAnTRMi.

- The A/D conversion of CG is interrupted immediately (ADCAnCTL1.ADCAnTRMi = 0).

  The A/D conversion of the interrupted channel is repeated after all pending A/D conversions of higher priority CGs have been finished.
- The A/D conversion of the current channel is completed before the higher priority CG is converted (ADCAnCTL1.ADCAnTRMi = 1).

  When the A/D conversions of *all* higher priority CGs have been finished, the interrupted A/D conversion is continued from the next channel.

ADCAnSTR2.ADCAnST[2:0] indicates the current conversion status.

Examples The following figures illustrate the different types of conversion interruption; CH3, CH9, and CH20 are assigned to CG0, CH5 and CH9 are assigned to CG2.

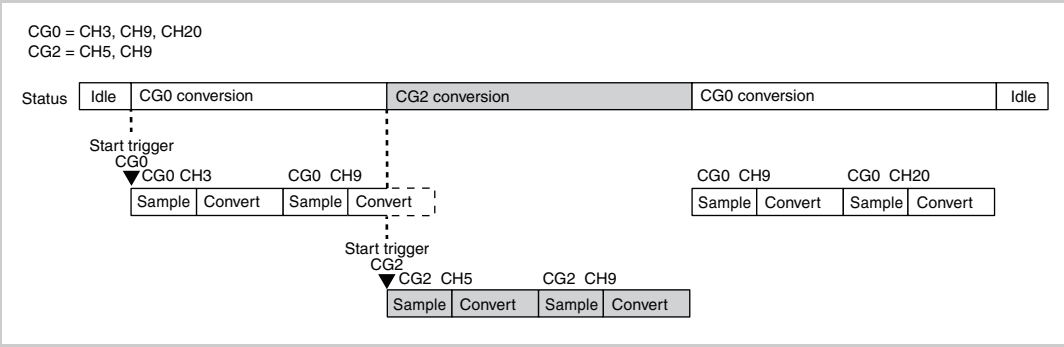


Figure 32-3 Immediate interrupt of A/D conversion of CG0 (ADCAnCTL1.ADCAnTRM0 is set to 0)

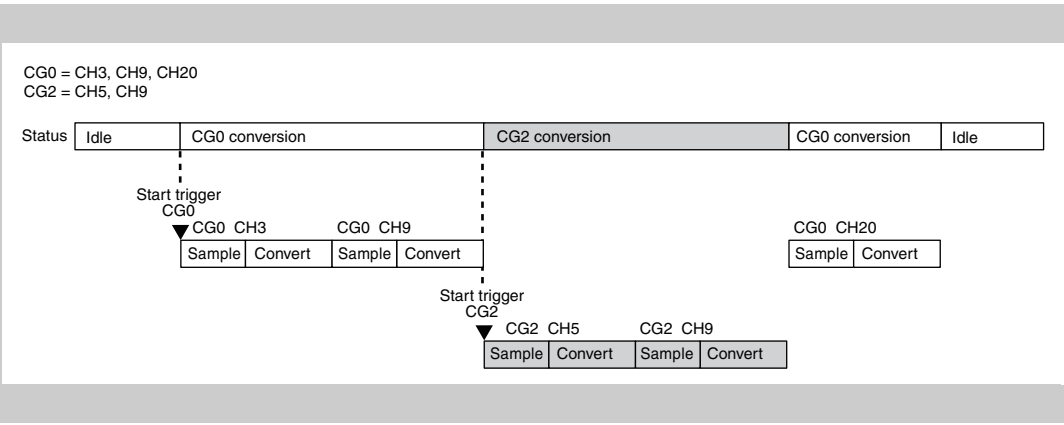


Figure 32-4 Wait until A/D conversion of current channel has been completed (ADCAnCTL1.ADCAnTRM0 is set to 1)

32.3.4 A/D conversion modes

The A/D converter provides the following two A/D conversion modes.

Mode	Operation	Channel group
One-shot conversion mode	Executes scan list conversion only once. In one-shot conversion mode, scan list conversion can be repeated the specified number of times (1 to 4 times).	CG0, CG1, CG2
Continuous conversion mode	Executes scan list conversion repeatedly.	CG0

Notes
1. A running A/D conversion is interrupted by an A/D conversion request for a higher priority CG and then is automatically continued when all requests of higher priority CGs have been finished (see *(1) "Order of A/D conversion" on page 2338*).
2. CG1 and CG2 operate in one-shot conversion mode regardless of the conversion mode setting. For CG0, the A/D conversion mode can be configured in ADCAnCTL1.ADCAn_MD0.

(1) One-shot conversion mode

In one-shot conversion mode, CGi scan list conversion is executed using the start trigger. The number of times scan list conversion is repeated can be specified for each CG from 1 to 4, by specifying ADCAnCTL0.ADCAnSCTi[1:0].

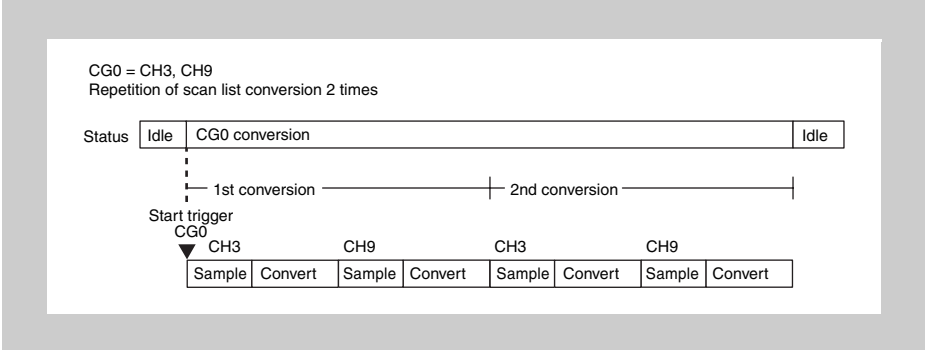


Figure 32-5 Repetition of scan list conversion 2 times in one-shot conversion mode

Operation when a start-before-conversion-end trigger is input

The A/D converter can hold one trigger for starting other conversion prior to the end of conversion of the same CG. Therefore, A/D conversion is performed continuously if one or more subsequent start triggers are input (the second and subsequent start triggers are ignored) before the A/D conversion for the CGi started by the first start trigger ends.

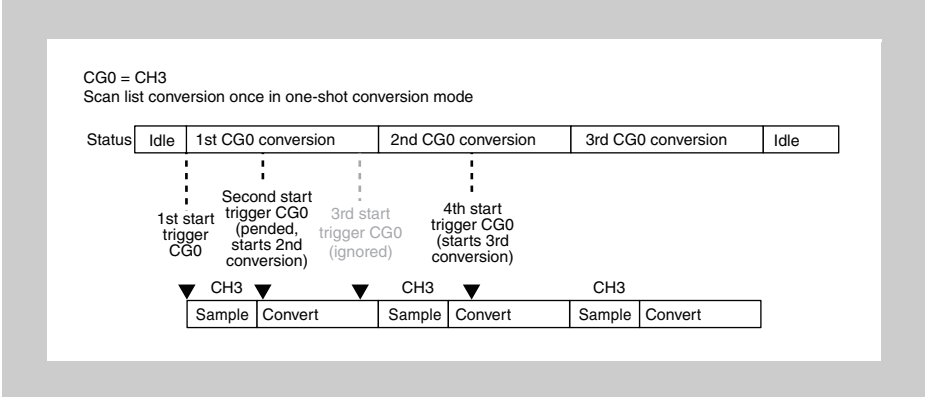


Figure 32-6 Operation when a start-before-conversion-end trigger is input

Caution During conversion of a high priority CG, start-before-conversion-end triggers for a lower priority CG are ignored. Reversely, a start-before-conversion end trigger for a low-priority CG that was input before the start of conversion of a high priority CG is held. When no start trigger is generated, a start-before-conversion-end trigger is accepted even during the conversion of a high priority CG.

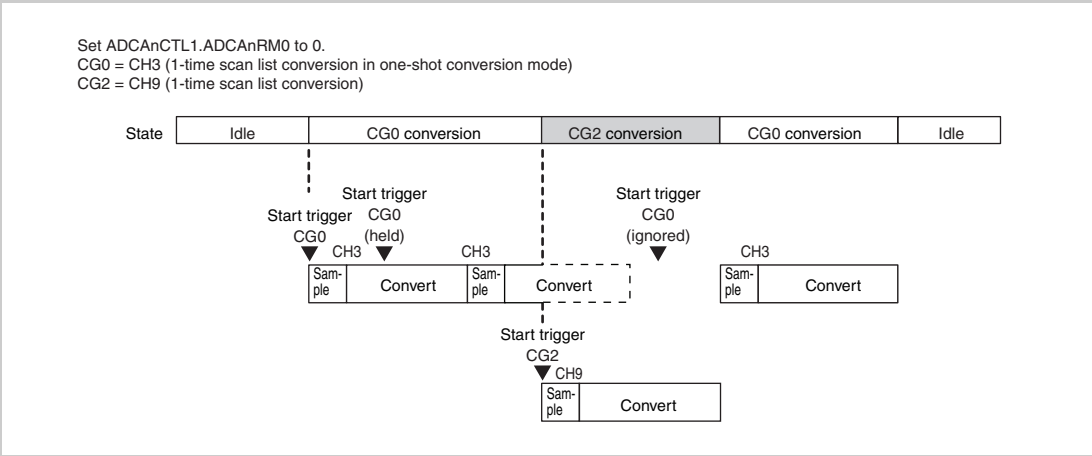


Figure 32-7 Operation when a start-before-conversion-end trigger is input (when start trigger for high priority CG is set)

(2) Continuous conversion mode

Continuous conversion mode can be used only for CG0 (ADCAnCTL1.ADCAn_MD0 = 1).

In continuous conversion mode, a start trigger causes the channels of CG0 to be sampled and converted repeatedly until a stop trigger is generated or another stop condition occurs (see *32.3.6 "Stopping A/D conversion (stop trigger)" on page 2344*).

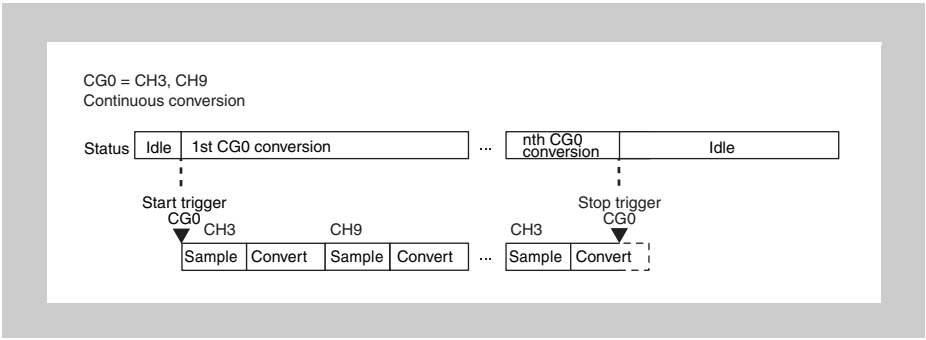


Figure 32-8 Continuous conversion mode

Caution After a stop trigger, the state becomes idle and sampling/conversion cannot be performed.

Note Additional start triggers for CG0 are ignored in continuous conversion mode.

32.3.5 Starting A/D conversion (start trigger)

A/D conversion can be started by a software or a hardware trigger which is specified in ADCAnCTL1.ADCAnMD1.

If A/D conversion is started for multiple CGs, the conversion order depends on the priority of the CGs (see *(1) "Order of A/D conversion" on page 2338*).

Notes

1. When no channels are assigned to CGi scan list (ADCAnCGi = 0000 0000_H), ADCAn ignores start triggers for that CGi.
2. In one-shot conversion mode, the A/D converter can hold only one start trigger.

   If one or more subsequent start triggers are input (the second and subsequent start triggers are ignored) before the started A/D conversion for the CGi ends, A/D conversion is performed continuously. (See *Figure 32-6 "Operation when a start-before-conversion-end trigger is input" on page 2341*).
3. In continuous conversion mode, subsequent start triggers started before the stop trigger is generated are ignored.

(1) Software start trigger

The A/D conversion of CGi is started by setting ADCAnTRGi.ADCAnSTTi to 1 – provided that the A/D converter is enabled (ADCAnCTL0.ADCAnCE is set to 1).

Timing example of software start trigger

The following figure shows the timing of a software start trigger with the following conditions:

- ADCAnTCLK = PCLK / 2 (ADCAnCTL1.ADCAnFR[3:0] = 000_B)

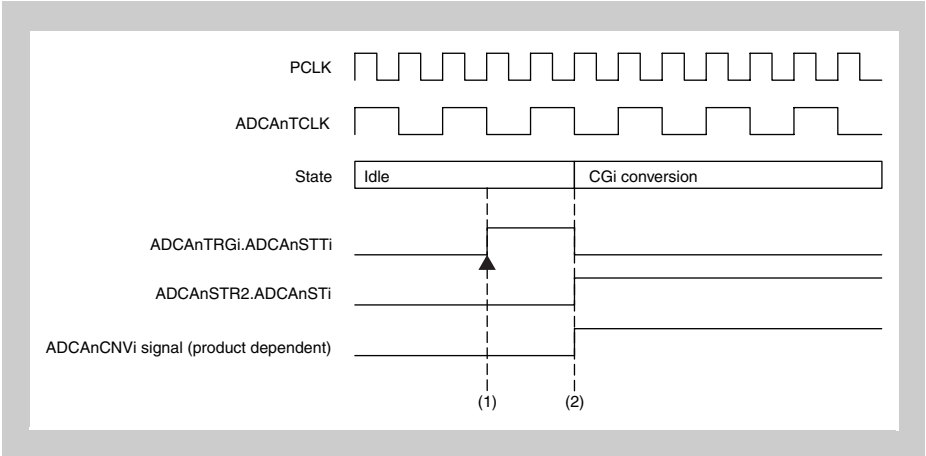


Figure 32-9 Software start trigger timing example

1. Software trigger for CGi is written.
2. A/D conversion starts at the next falling edge of ADCAnTCLK.

   The status signal ADCAnCNVi becomes active (product dependent) and ADCAnSTR2.ADCAnSTi is set, indicating that A/D conversion for CGi is running.

(2) Hardware start trigger

The A/D conversion for CGi is started upon detection of the valid edge of the ADCAnTTRGi signal, provided that the A/D converter is enabled (ADCAnCTL0.ADCAnCE = 1) and hardware trigger mode is set (ADCAnCTL1.ADCAnMD1 = 1).

The valid edge is specified in ADCAnCTL1.ADCAnTiETS[1:0] for every CG individually.

Hardware trigger expansion (product dependent)

If this microcontroller supports a hardware trigger expansion, up to 16 hardware trigger sources can be specified for each ADCATTRGi signal input. ADCAnTSELi specifies the input signals to be used as the ADCATTRGi signal.

Note See “Connection destination of hardware trigger” in the first section of this chapter for the connection destination of the hardware start trigger.

Timing of hardware start trigger

The A/D converter starts A/D conversion upon detection of the valid edge of the ADCAnTTRGi signal.

The following figure shows the timing of a hardware start trigger with the following conditions:

- ADCAnTCLK = PCLK / 2 (ADCAnCTL1.ADCAnFR[3:0] = 000_B)
- Valid edge of ADCAnTTRGi is rising edge (ADCAnCTL1.ADCAnTiETS[1:0] = 01_B)

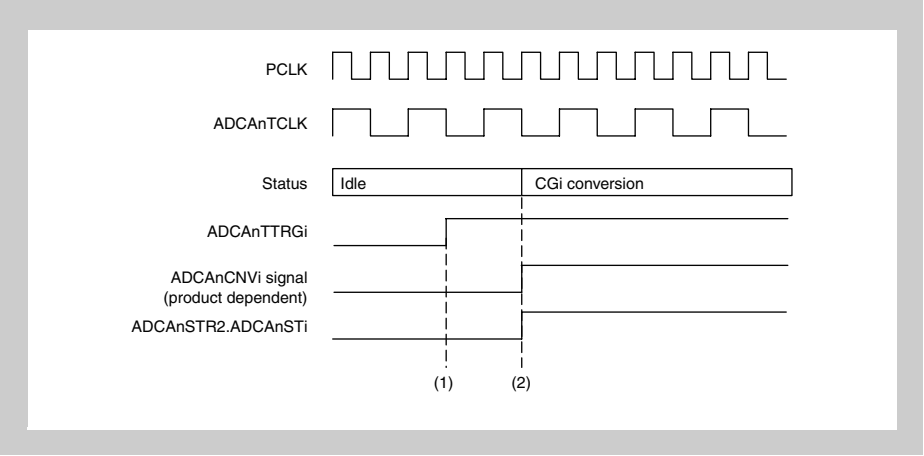


Figure 32-10 Hardware start trigger timing

1. Input signal ADCAnTTRGi rises.
2. A/D conversion starts at the next falling edge of ADCAnTCLK.

   The status signal ADCAnCNVi becomes active (product dependent) and ADCAnSTR2.ADCAnSTi is set, indicating that A/D conversion for CGi is running.

32.3.6 Stopping A/D conversion (stop trigger)

(1) Stop trigger

Setting the stop trigger bit for CGi to 1 (ADCAnTRG4+i.ADCAnSPi = 1) stops the A/D conversion for the CGi. If the stop trigger is generated before the A/D conversion ends, the A/D conversion end interrupt signal INTADCAnTi is not generated and the A/D conversion result register is not updated. If the start trigger is generated again after the A/D conversion is stopped by a stop trigger, scan list conversion is executed from the beginning.

Follow the procedure below when using the hardware start trigger.

1. Stop hardware start trigger generation.
2. Set the stop trigger bit (ADCAnTRG4+i.ADCAnSPi) to 1.
3. Check the status of ADCAnSTR2.ADCAnSTi.

If the above procedure is not followed, the hardware start trigger and the stop trigger may conflict with each other and A/D conversion may not stop.

Timing of stop trigger

1. Stop trigger for CGi is written.
2. A/D conversion of CGi stops at the next falling edge of ADCAnTCLK.

   ADCAnSTR2.ADCAnSTi is cleared, indicating that the A/D conversion for CGi is stopped.
   If the digital value of ADCAnIm can already be used, the operation is as follows:
   - All A/D conversion result registers are updated.
   - The conversion end interrupt INTADCAnTi is generated as configured in ADCAnIOCi (see *32.3.10 “Interrupt generation” on page 2349*).
   - If configured in ADCAnCTL2, the A/D conversion result is checked to see whether it is in the specified value range (see *32.3.12 “Result check functions” on page 2353*).

The A/D converter proceeds with pending A/D conversion requests of other CGs - if there are any.

The following figures show the timing of a stop trigger with the following conditions:

- ADCAnTCLK = PCLK / 2 (ADCAnCTL1.ADCAnFR[3:0] = 000_B)
- The A/D conversion end interrupt INTADCAnTi is generated at the end of the A/D conversion for CGi (ADCAnIOCi = 0000 0000_H)

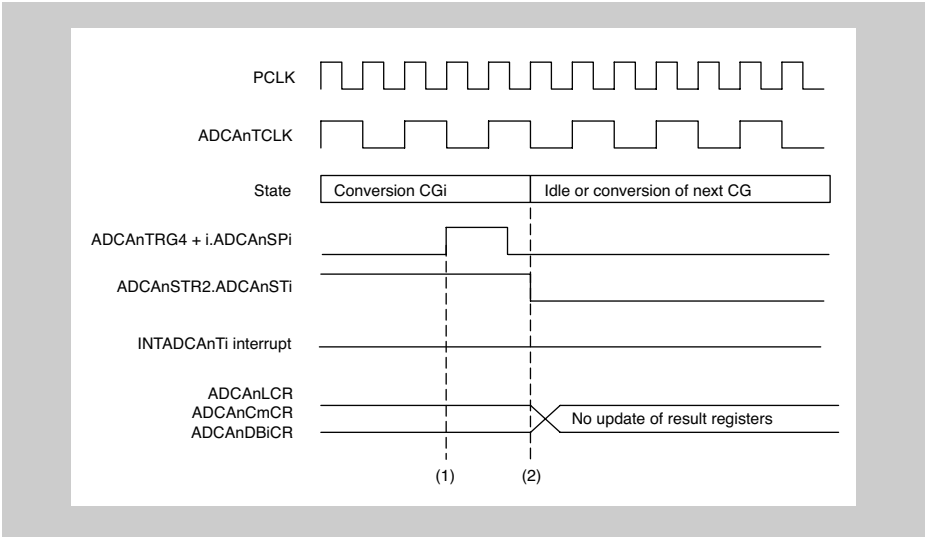


Figure 32-11 Stop trigger timing (if generated before A/D conversion end)

1. The stop trigger (ADCAnTRG4+i.ADCAnSPi) is set to 1.
2. The status bit (ADCAnSTR2.ADCAnSTi) is cleared.

   The A/D conversion end interrupt INTADCAnTi is not generated and the A/D conversion result register is not updated.

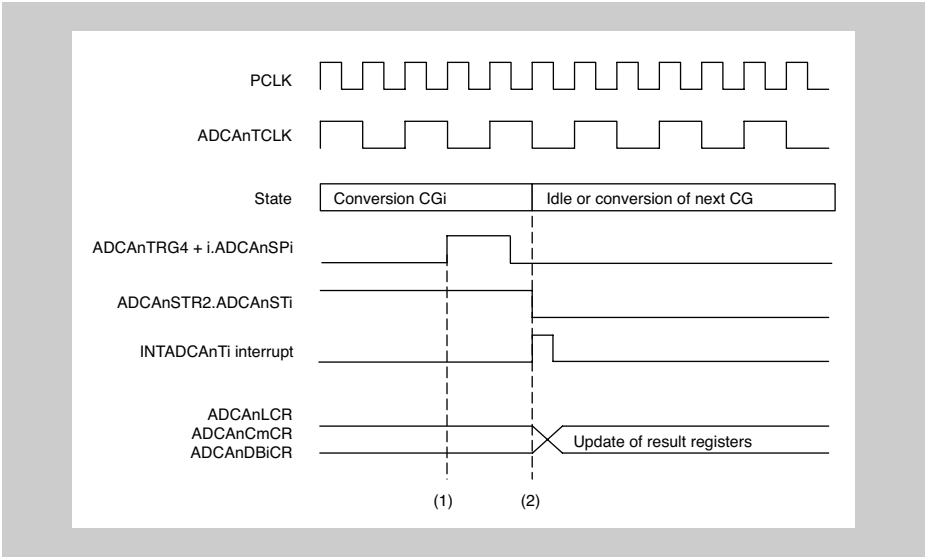


Figure 32-12 Stop trigger timing (if generated after A/D conversion end)

1. The stop trigger (ADCAnTRG4+i.ADCAnSPi) is set to 1.
2. The status bit (ADCAnSTR2.ADCAnSTi) is cleared.

The A/D conversion end interrupt (INTADCAnTi) is generated and the A/D conversion result register is updated.

(2) Other stop conditions

In addition to the software stop trigger, A/D conversion is stopped with the following conditions:

- When A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0)

32.3.7 Standby mode (product dependent)

Standby mode is entered as described in “Conditions for transition to standby mode” in the first section of this chapter.

The A/D converter is automatically disabled (ADCAnCTL0.ADCAnCE = 0).

To leave standby mode:

1. Release related system standby mode.
2. Enable the A/D converter by setting ADCAnCTL1.ADCAnCE to 1.

Note After standby mode is released, a start trigger is accepted but conversion does not start until the stabilization time elapses (stabilization counter ADCAnCNT = 00_H). See *32.3.17 “Stabilization control” on page 2370* for details.

32.3.8 Pausing and resuming A/D conversion (ADCHALT mode) (product dependent)

The A/D converter allows the A/D conversion (of all CGs) to be paused/halted (ADCHALT mode).

Procedure:

1. Set ADCAnTRG3.ADCAnSTT3 to 1 to go into ADCHALT mode. (See *(1) “Order of A/D conversion” on page 2338* for details on the halt operation.)
   – Start triggers are ignored in ADCHALT mode.
   – The internal circuits are stopped and the consumption power is reduced by setting the sampling clock ADCAnTCLK to low level.
   – The analog input pins ADCAnIm can be used for other functions in ADCHALT mode.
2. Set ADCAnTRG7.ADCAnSP3 to 1 to release ADCHALT mode and start A/D conversion.

Note ADCHALT has the highest priority and overrules all CGi conversions.

32.3.9 Resolution, sampling and conversion times

The total conversion time comprises sampling time and A/D conversion time.

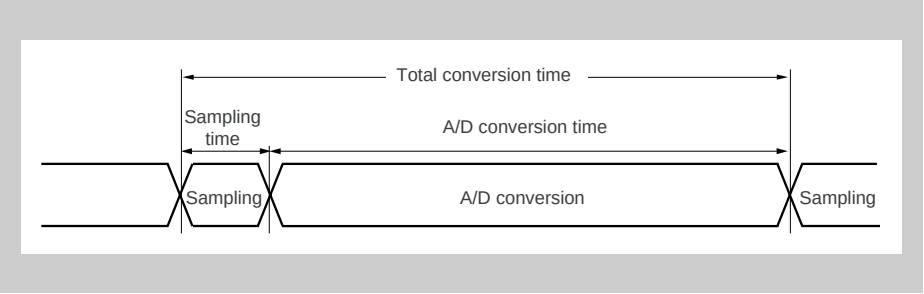


Figure 32-13 Total conversion time

- The *sampling time* is the time of connecting an analog input voltage to the common sample & hold circuit. The *A/D conversion time* is the time required to obtain one digital value out of an analog input voltage.
  The *A/D conversion time*, and thus the *total conversion time,* depends on the conversion resolution:

<R> **Table 32-9 Sampling and conversion times (product dependent)**

Sampling mode (product dependent)	Conversion resolution	Sampling time	Total conversion time
A	10 bits (ADCAnCTL1.ADCAnCTYP is set)	6.5 clocks (ADCAnTCLK)	18 clocks (ADCAnTCLK)
	12 bits (ADCAnCTL1.ADCAnCTYP is cleared)	6.5 clocks (ADCAnTCLK)	20 clocks (ADCAnTCLK)

Notes
1. The total conversion time becomes 1 clock (ADCAnTCLK) longer when the discharge function is enabled (ADCAnCTL1.ADCAnDISC = 1).
   See *32.3.15 “Discharge function” on page 2369* for details.
2. The total conversion time becomes 4 clocks (ADCAnTCLK) longer when the power amplifier function is enabled (ADCAnTCLK = 1).
   See *32.3.16 “Buffer amplifier function” on page 2369* for details.

Table 32-10 Conversion times for sampling mode A (product dependent)

Resolution: 12 bits, discharge function: enabled, amplifier power: on
(conversion time for 25 clocks (ADCAnTCLK) [µs])

12-bit resolution		PCLK <MHz> (product dependent)							
ADCAnFR3 to ADCAnFR0	Division ratio	16	20	24	32	40	48	64	80
0000_B	PCLK/2	3.125	2.500	2.083	1.563	a	a	a	a
0001_B	PCLK/3	4.688	3.750	3.125	2.344	1.875	1.563	a	a
0010_B	PCLK/4	6.250	5.000	4.167	3.125	2.500	2.083	1.563	a
0011_B	PCLK/5	7.813	6.250	5.208	3.906	3.125	2.604	1.953	1.563
0100_B	PCLK/6	9.375	7.500	6.250	4.688	3.750	3.125	2.344	1.875
0110_B	PCLK/8	a	10.000	8.333	6.250	5.000	4.167	3.125	2.500
1000_B	PCLK/10	a	a	a	7.813	6.250	5.208	3.906	3.125
1010_B	PCLK/12	a	a	a	9.375	7.500	6.250	4.688	3.750
1100_B	PCLK/14	a	a	a	a	8.750	7.292	5.469	4.375
1110_B	PCLK/16	a	a	a	a	10.000	8.333	6.250	5.000

a) Setting prohibited

Resolution: 10 bits, discharge function: enabled, amplifier power: on
(conversion time for 23 clocks (ADCAnTCLK) [µs])

10-bit resolution		PCLK <MHz> (product dependent)							
ADCAnFR3 to ADCAnFR0	Division ratio	16	20	24	32	40	48	64	80
0000_B	PCLK/2	2.875	2.300	1.917	a	a	a	a	a
0001_B	PCLK/3	4.313	3.450	2.875	2.156	1.725	a	a	a
0010_B	PCLK/4	5.750	4.600	3.833	2.875	2.300	1.917	a	a
0011_B	PCLK/5	7.188	5.750	4.792	3.594	2.875	2.396	1.797	a
0100_B	PCLK/6	8.625	6.900	5.750	4.313	3.450	2.875	2.156	1.725
0110_B	PCLK/8	a	9.200	7.667	5.750	4.600	3.833	2.875	2.300
1000_B	PCLK/10	a	a	9.583	7.188	5.750	4.792	3.594	2.875
1010_B	PCLK/12	a	a	a	8.625	6.900	5.750	4.313	3.450
1100_B	PCLK/14	a	a	a	a	8.050	6.708	5.031	4.025
1110_B	PCLK/16	a	a	a	a	9.200	7.667	5.750	4.600

a) Setting prohibited

Note The above listed conversion times do not include the overhead time required for the processing by the A/D controller. The following times are required for the overhead time.

One clock (ADCAnTCLK) for accepting a conversion start request

One clock (ADCAnTCLK) for executing initialization before conversion start

One clock (ADCAnTCLK) for saving the conversion result

<R>

32.3.10 Interrupt generation

(1) A/D conversion end interrupt INTADCAnTi

The INTADCAnTi interrupt indicates that the new A/D conversion result is saved to the conversion result register.

An A/D conversion end interrupt is generated upon completion of the A/D conversion for any channel of CGi specified in the ADCAnIOCi register.

If nothing else is configured (ADCAn_IOCi = 0000 0000_H), INTADCAnTi is generated at the end of the A/D conversion of CGi.

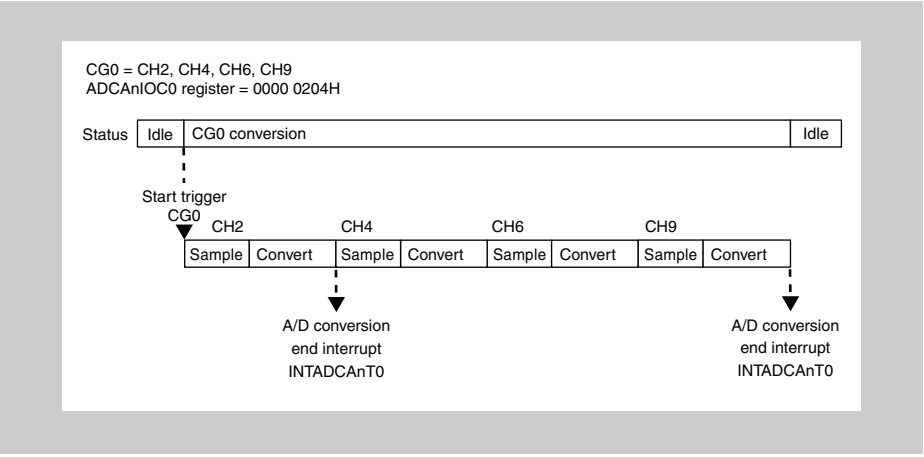


Figure 32-14 Generation of A/D conversion end interrupt INTADCAnTi

Notes

1. ADCAnIOCi can be written at any time even when the A/D converter is enabled (ADCAnCTL0.ADCAnCE = 1). The new value takes effect after the current A/D conversion of CGi has been completed.
2. ADCAnIOCi is associated with ADCAnCGi and their buffer registers should be updated simultaneously. As the update time depends on writing ADCAnCGi, always write ADCAnIOCi before ADCAnCGi if you want to change the interrupt generation for a CG.

(2) Error interrupt INTADCAnTERR

The interrupt INTADCAnTERR is generated in the following cases:

- The result of the A/D conversion of a specified channel is out of the specified range, when the conversion result upper/lower limit compare function is enabled.

  See *(3) "Conversion result upper/lower limit compare function" on page 2354* for details.

- The A/D conversion result in the ADCAnLCR, ADCAnDBiCR, or ADCAnCmCR register is overwritten before it is read.

  The generation of the INTADCAnTERR error interrupt upon register overwrite can be controlled for each register, by setting ADCAnCTL0.ADCAnOEM[4:0].
  See *(1) "Conversion result overwrite check function" on page 2353* for details.

32.3.11 Storage of A/D conversion result

(1) A/D conversion result registers

The A/D conversion result is stored in the following registers:

- ADCAnLCR register

  This register stores the latest A/D conversion result.

- ADCAnDBiCR register

  This register stores the latest A/D conversion result for CGi.

- ADCAnCmCR register

  This register allows you to continuously read the A/D conversion results for channel m.

These registers store the digital value for the sampled analog input voltage in bits 15 to 0. Each register also stores the status flag of the A/D conversion result (see *32.3.12 “Result check functions” on page 2353*).

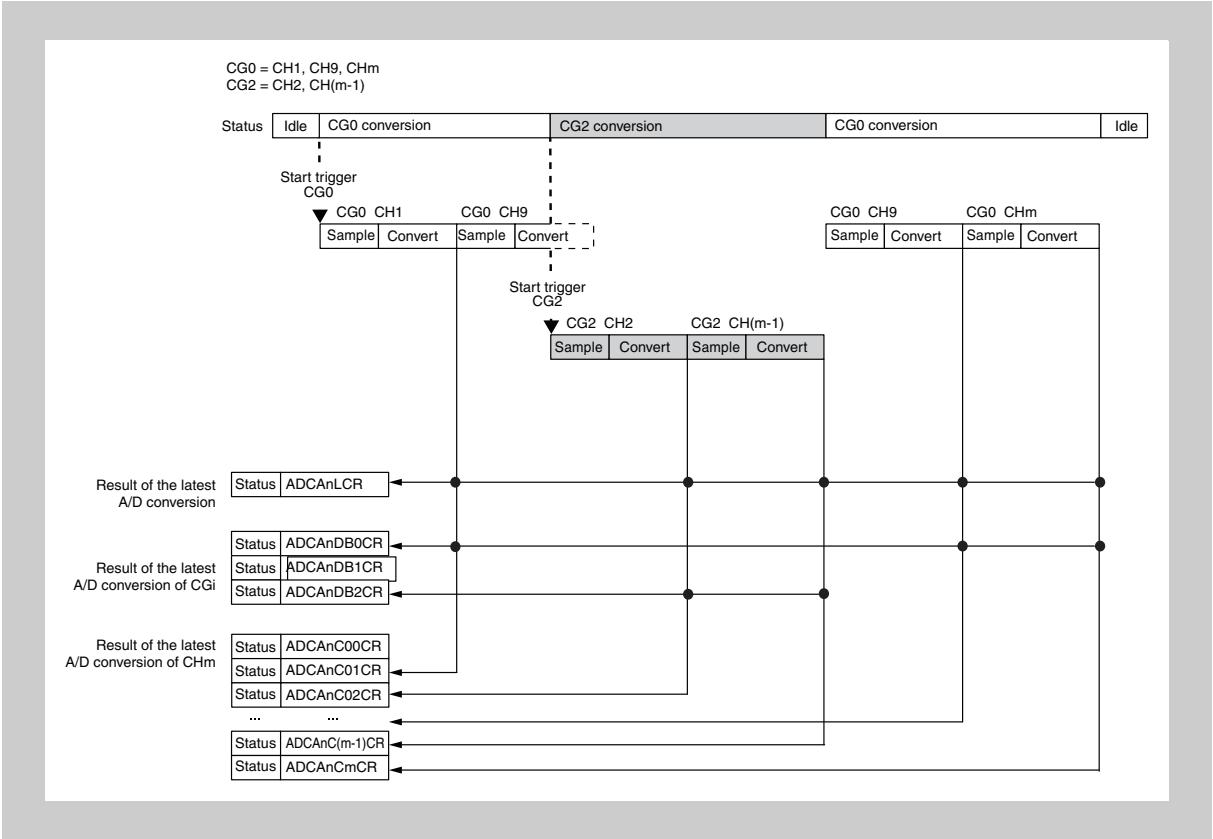


Figure 32-15 Storage of A/D conversion result

(2) Configuration related to A/D conversion result storage

(a) Bit position specification of A/D conversion result

ADCAn_CTL1.ADCAn_CRAC specifies whether the 12-bit or 10-bit A/D conversion result is right aligned (ADCAnCRAC = 0) or left aligned (ADCAnCRAC = 1).

(b) Conversion result read & clear function

Whether to keep or clear the value of the A/D conversion result register ADCAnCmCR after it is read can be specified by ADCAnCTL1.ADCAnRCL.

(3) Relationship between analog input voltage and A/D conversion result

The relationship between the analog input voltage input to the analog input pin (ADCAnIm) and the A/D conversion results (the values of the ADCAnLCR[15:00] bits, ADCAnCmCR[15:00] bits, and ADCAnDBiCR[15:00] bits) is expressed as follows:

$$\text{A/D conversion result value} = \text{INT}\left(\frac{V_{IAN} - AV_{REFMn}}{AV_{REFPn} - AV_{REFMn}} \times 2^k + 0.5\right)$$

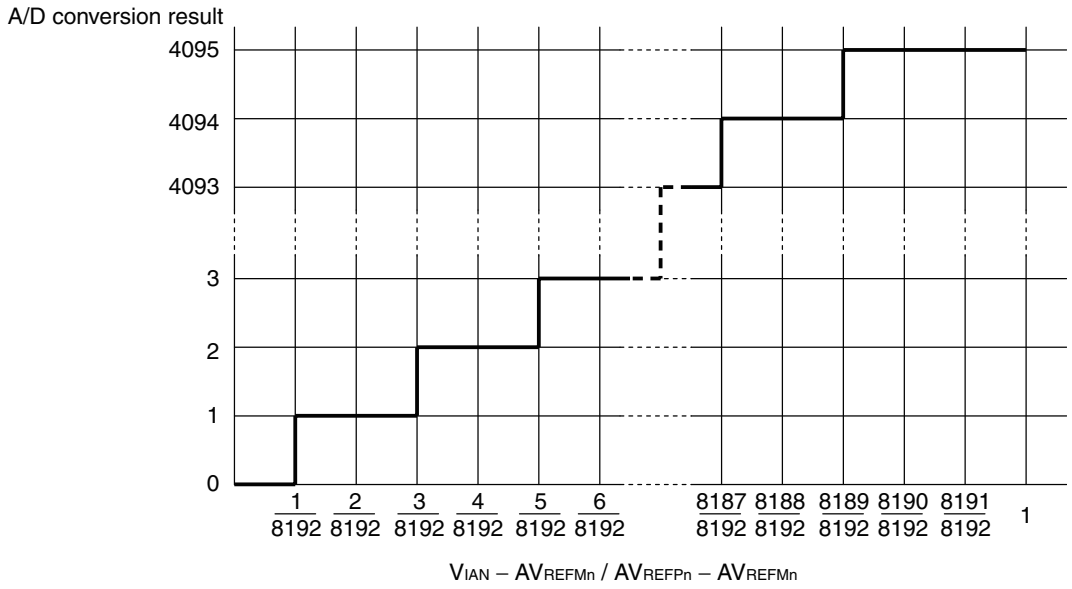
or

$$(\text{A/D conversion result} - 0.5) \times \frac{AV_{REFPn} - AV_{REFMn}}{2^k} \leq V_{IAN} - AV_{REFMn} < (\text{A/D conversion result} + 0.5) \times \frac{AV_{REFPn} - AV_{REFMn}}{2^k}$$

INT():	Function that returns the integer part of the value in parentheses
V_{IAN}:	Analog input voltage
AV_{REFPn}:	AV_{REFPn} pin voltage
AV_{REFMn}:	AV_{REFMn} pin voltage
A/D conversion result:	Values of the ADCAnLCR[15:00], ADCAnCmCR[15:00], and ADCAnDBiCR[15:00] bits
k:	Resolution

*Figure 32-16 “Relationship between analog input voltage and A/D conversion result”*shows the relationship between the analog input voltage and A/D conversion result.

(i) Conversion characteristics of 12-bit A/D converter (ADCAnCTL1.ADCAnCTYP = 0)



(2) Conversion characteristics of 10-bit A/D converter (ADCAnCTL1.ADCAnCTYP = 1)

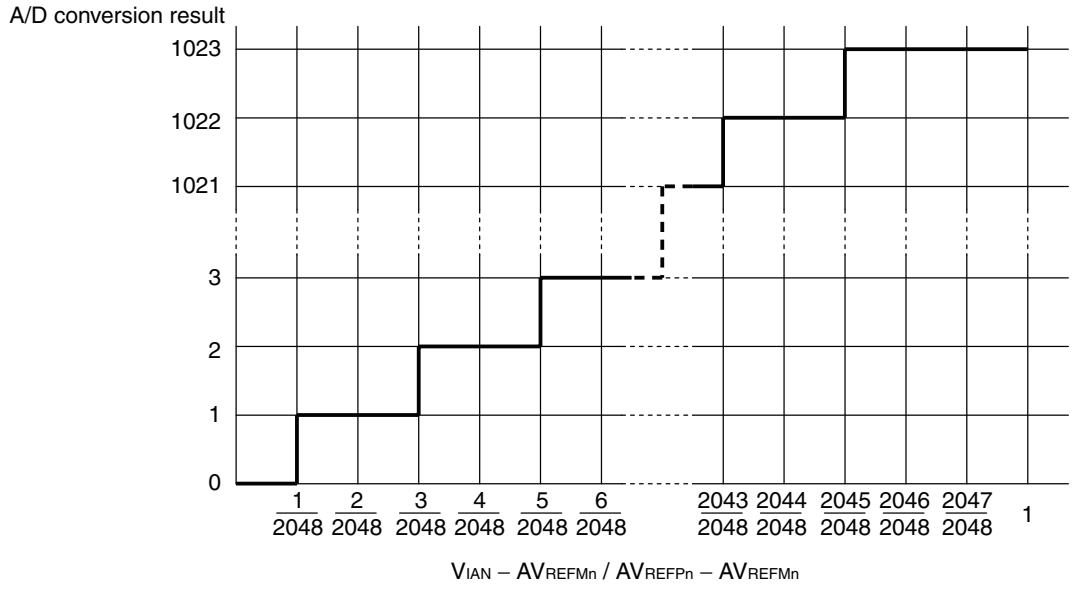


Figure 32-16 Relationship between analog input voltage and A/D conversion result

32.3.12 Result check functions

ADCAn allows checking of the A/D conversion result with the following functions.

- Conversion result overwrite check function
- Conversion result read flag result
- Conversion result upper/lower limit compare function

(1) Conversion result overwrite check function

The ADCA allows to check if an A/D conversion result has been overwritten before it was read.

Error flags

The A/D conversion result register has the following overwrite error flags.

ADCAnLCR.ADCAnLER1

ADCAnDBiCR.ADCAnDBiER1

ADCAnCmCR.ADCAnCmER1

For example, if the A/D conversion result stored in the ADCAnCmCR register is overwritten before it is read, ADCAnCmCR.ADCAnCmER1 is set.

The same applies for the ADCAnLCR and ADCAnDBiCR registers.

The value of ADCAnCmCR.ADCAnCmER1 is reflected to ADCAnSTR1.ADCAnOWEm.

Error interrupt

If the A/D conversion results in the ADCAnLCR, ADCAnDBiCR, and ADCAnCmCR registers are overwritten before they were read, the error interrupt INTADCAnTERR is generated.

Conversion result registers that should not to be read can be masked by setting ADCAnCTL0.ADCAnOEM[4:0].

Example:

- CH7 is assigned to CG1.
- Set ADCAnCTL0.ADCAnOEM[4:0] to 00000B so that the INTADCAnTERR error interrupt is generated when the A/D conversion results in the ADCAnLCR, ADCAnDBiCR, and ADCAnCmCR registers are overwritten before they are read.

(2) Conversion result read flag function

Whether the A/D conversion result is read or whether it is new and not yet read, can be checked.

Status flag

The following update status flags are provided for the A/D conversion result registers.

ADCAnLCR.ADCAnLUR

ADCAnDBiCR.ADCAnDBiUR

ADCAnCmCR.ADCAnCmUR

If these flags are set to 1, the A/D conversion result is new.

The update status flags are cleared after they are read.

(3) Conversion result upper/lower limit compare function

Whether the A/D conversion result is within the setting range can be checked.

This function can be enabled and disabled for each channel with the ADCAnCTL2 register.

The conversion result of a channel for which this function is enabled is compared with the lower limit value (ADCAnLL register) and the upper limit value (ADCAnUL register) that are set beforehand.

Error flags

If the A/D conversion result of a specified channel either is lower than the lower limit value or higher than the upper limit value, the ADCAnSTR0.ADCAnRCE error flag corresponding to that channel is set.

The ADCAnSTR0 register indicates the error status for the latest A/D conversion result upper/lower limit comparison of each channel. Which A/D conversion result is out of the setting range can be checked with the ADCAnSTR0 register.

The value of the result check error flag ADCAnSTR0.ADCAnRCE is reflected to ADCAnCmCR.ADCAnCmER0.

Error interrupt

If the A/D conversion result for the specified channel is out of the setting range, the INTADCAnTERR error interrupt is generated.

32.3.13 Self-diagnosis functions

The following self-diagnosis functions can be used to verify that the ADCAn works properly:

- *(1) "Diagnosis for A/D conversion circuit" on page 2356*
- *(2) "Diagnosis for channel multiplexer" on page 2357*
- *(3) "Diagnosis for open pins" on page 2358*
- *(4) "Channel S/H circuit diagnosis (product dependent)" on page 2359*

The figure outlines the self-diagnosis functions which are explained in detail in the following chapters.

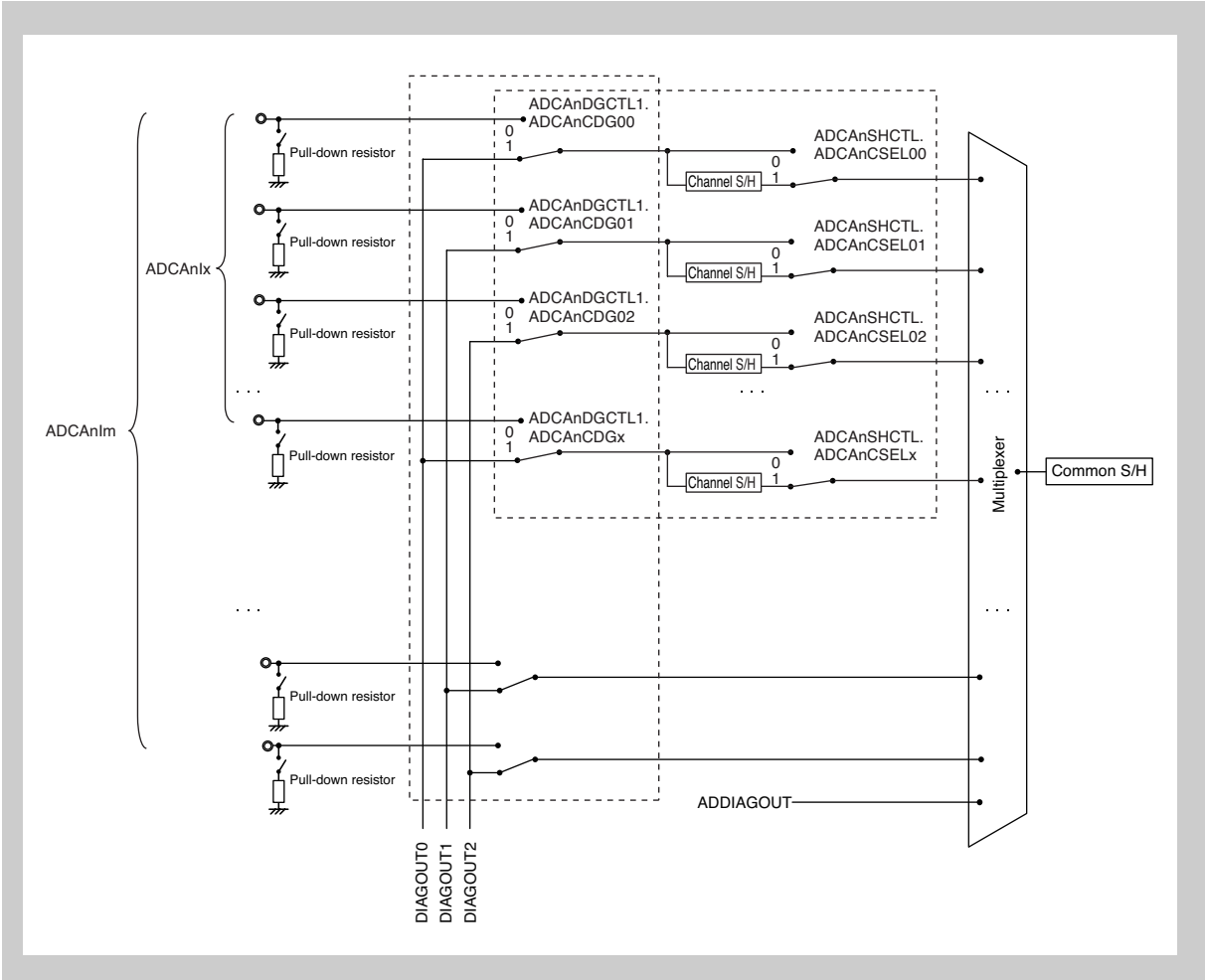


Figure 32-17 Outline of self-diagnosis functions

Note The parts enclosed in dotted lines are product dependent functions.

The cases when the various diagnosis functions can be set are listed in the table below.

Table 32-11 Setting of self-diagnosis functions

Self-diagnostic function	ADCAnCE = 0	ADCAnCE = 1
A/D conversion circuit diagnosis	Can be set	Can be set
Channel multiplexer diagnosis)	Can be set	Cannot be set
Open pin diagnosis	Can be set	Cannot be set
Channel S/H circuit diagnosis (product dependent)	Can be set	Can be set

(1) Diagnosis for A/D conversion circuit

The operation of the A/D conversion circuit can be diagnosed.

The diagnosis of the A/D conversion circuit can be performed during normal A/D conversion. Following the end of A/D conversion of CG0, the ADDIAGOUT reference voltage signal is converted. If this diagnosis A/D conversion result differs greatly from the expected value, a hardware anomaly or a malfunction may occur.

The diagnostic A/D conversion is enabled by setting ADCAnCG0.ADCAnDIAG to 1.

Note The A/D conversion circuit diagnosis is available for CG0 only.

The diagnostic A/D conversion is started after the A/D conversion of the last channel of CG0 has been completed:

- The A/D conversion results of CG0 are stored in the “normal” A/D conversion result registers (see *(1) “A/D conversion result registers” on page 2350*).
- The result of the diagnostic A/D conversion is stored in ADCAnDGCR.

Diagnosis procedure

1. Switch the power of the ADCAn on by setting ADCAnCTL1.ADCAnGPS to 1.
2. Configure CG0 and the A/D conversion as follows:
   - Ensure that ADCAnCG0.ADCAnDIAG is set to 1 to enable the diagnostic A/D conversion of the reference voltage.
     For example, write 8000 000E$_H$ to first convert the analog input voltages of CH0, CH1, and CH2, and than the reference voltage ADDIAGOUT for diagnostic purposes.
   - Set ADCAnI0C0.ADCAnCG0IDG to 1 to generate the A/D conversion end interrupt INTADCAnT0 on finishing the diagnostic A/D conversion.
3. Specify the reference voltage signal ADDIAGOUT in ADCAnDGCTL0.ADCAnPSEL[2:0].
   For example, set ADCAnDGCTL0.ADCAnPSEL[2:0] to 010$_B$ to apply the reference voltage 1/2 AV_{DD}.
4. Enable the ADCAn by setting ADCAnCTL0.ADCAnCE to 1.
5. Generate software or hardware start triggers to start the A/D conversion.
6. When the A/D conversion end interrupt INTADCAnT0 is generated, read the A/D conversion results of the diagnostic A/D conversion in ADCAnDGCR.

ADCAnDGCTL0.ADCAnPSEL[2:0] can be written even during A/D conversion.

The operation when ADCAnDGCTL0.ADCAnPSEL[2:0] are written during A/D conversion is shown in the following figure.

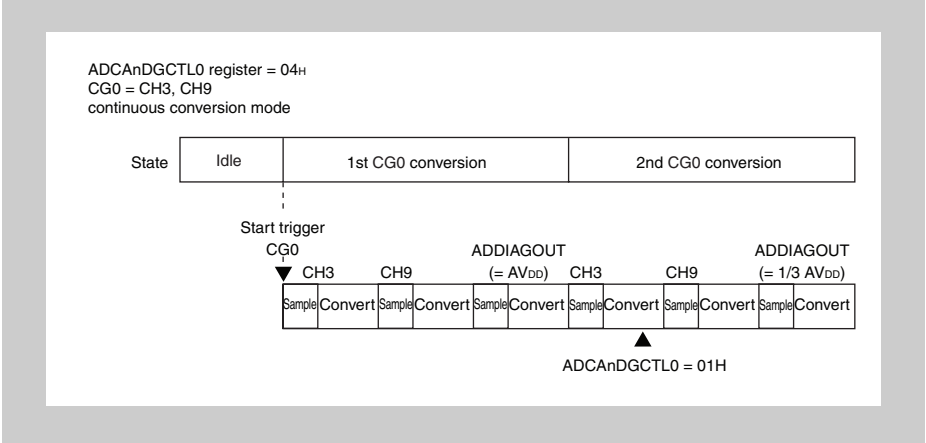


Figure 32-18 Write during A/D conversion

Note The value set with ADCAnDGCTL0.ADCAnPSEL[2:0] is applied upon completion of the conversion of the current channel. Therefore, set the reference voltage of the next diagnosis A/D conversion before that diagnosis A/D conversion starts.

(2) Diagnosis for channel multiplexer

Diagnosis of channel switching and selection can be performed.

Each channel is permanently assigned to one of three reference voltages. The values of the reference voltages can be altered using ADCAnDGCTL0.ADCAnPSEL[2:0].

Table 32-12 Channel assignment of reference voltages

Reference voltage	Assigned channels
DIAGOUT0	21, 18, 15, 12, 9, 6, 3, 0
DIAGOUT1	22, 19, 16, 13, 10, 7, 4, 1
DIAGOUT2	23, 20, 17, 14, 11, 8, 5, 2

To diagnose the channel multiplexer, different reference voltages can be input to the channels.

Diagnosis procedure

1. Switch the power of the ADCAn on by setting ADCAnCTL1.ADCAnGPS to 1.
2. Specify the channels of CG0 in ADCAnCG0.

   For example, write 0000 000E_H to use CH0, CH1, and CH2 for diagnosis.
3. Configure ADCAnI0C0 to generate the A/D conversion end interrupt INTADCAnTi at the end of all A/D conversions for CG0.
4. Configure the other A/D conversion settings as desired.
5. Specify the reference voltages in ADCAnDGCTL0.ADCAnPSEL[2:0].

   For example, set ADCAnDGCTL0.ADCAnPSEL[2:0] to 010_B to use the following reference voltages:

– DIAGOUT0 = 1/2 AV_{DD}
– DIAGOUT1 = 2/3 AV_{DD}
– DIAGOUT2 = 1/3 AV_{DD}

6. Configure to which channels of CG0 a reference voltage is applied (instead of the analog input voltage ADCAnIm) in ADCAnDGCTL1. For example, set ADCAnDGCTL1 to 0000 0006_H:
   – DIAGOUT1 (2/3 $AV_{DD)}$ will be applied to CH1.
   – DIAGOUT2 (1/3 $AV_{DD)}$ will be applied to CH2.
   – Analog input voltage will be applied to CH3.
7. Enable the ADCAN by setting ADCAnCTL0.ADCAnCE to 1.
8. Generate software or hardware start triggers to start the A/D conversion.
9. When receiving the A/D conversion end interrupt INTADCAnT0, read the A/D conversion results of CG0.

Caution When ADCAnCTL0.ADCAnCE = 1, changing the setting of the ADCAnDGCTL1 register is prohibited.

(3) Diagnosis for open pins

The correct A/D conversion result cannot be obtained when the input pins are open.

The internal pull-down resistors can be connected to test the analog inputs ADCAnIm.

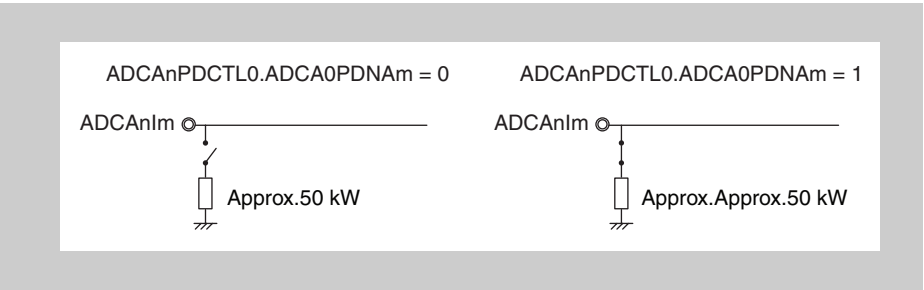


Figure 32-19 Internal pull-down resistors

When an internal pull-down resistor is connected to the ADCAnIm analog input (ADCAnPDCTL0.ADCAnPDNAm = 1) and ADCAnIm is open, the A/D conversion result declines to almost 0 V.

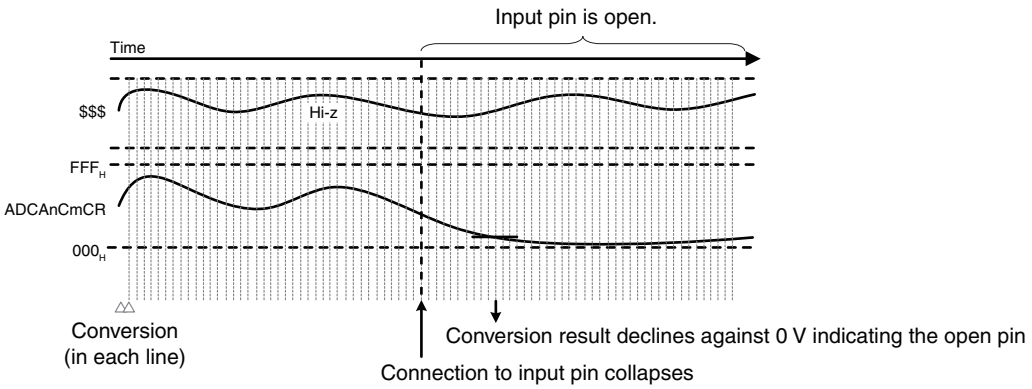


Figure 32-20 Detection of an open input pin

Notes

1. When the analog input ADCAnIm is smaller than 0.2 V (T.B.D.), the pin open state cannot be determined (see the Data Sheet).
2. The internal pull-down resistors must not be connected during normal A/D conversion operation. Connecting internal pull-down resistors may lead to a drop in the input voltage and may result in imprecise A/D conversion results.

Diagnosis procedure

To diagnose open input pins:

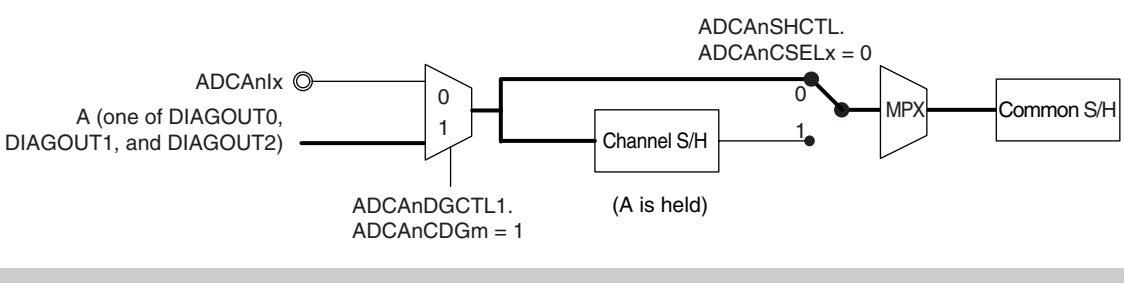
1. Configure the CG and A/D conversion settings as usual.
2. Configure the internal pull-down resistors to be connected in ADCAnPDCTL0.
3. Start A/D conversions several times (T.B.D.).
4. Monitor the channel's A/D conversion results and check if any result declines to almost 0 V.

(4) Channel S/H circuit diagnosis (product dependent)

See *32.3.14 "Channel S/H function (product dependent)"* for details about the channel S/H function.

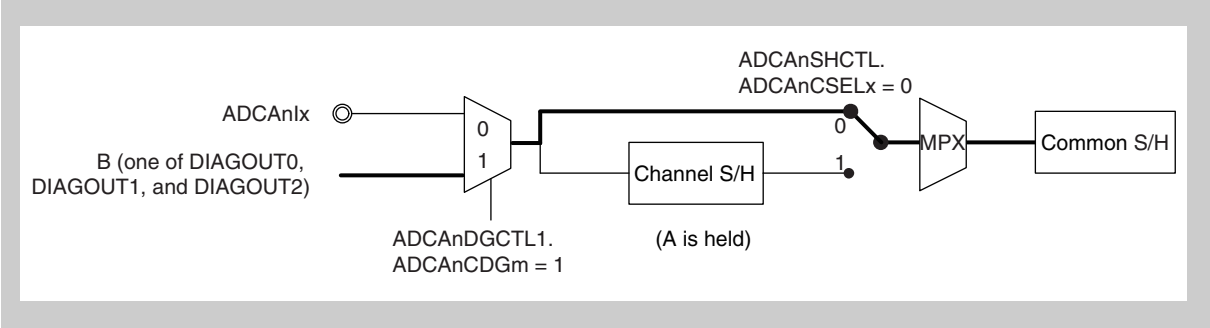
The channel S/H circuit can be diagnosed. An overview is given below.

1. Use one of the reference voltage signals DIAGOUT0 to DIAGOUT2.
2. Set voltage "A" for the selected reference voltage signal (DIAGOUT0, DIAGOUT1, or DIAGOUT2). Conversion is performed by holding "A" in the channel S/H circuit, without using the channel S/H circuit.



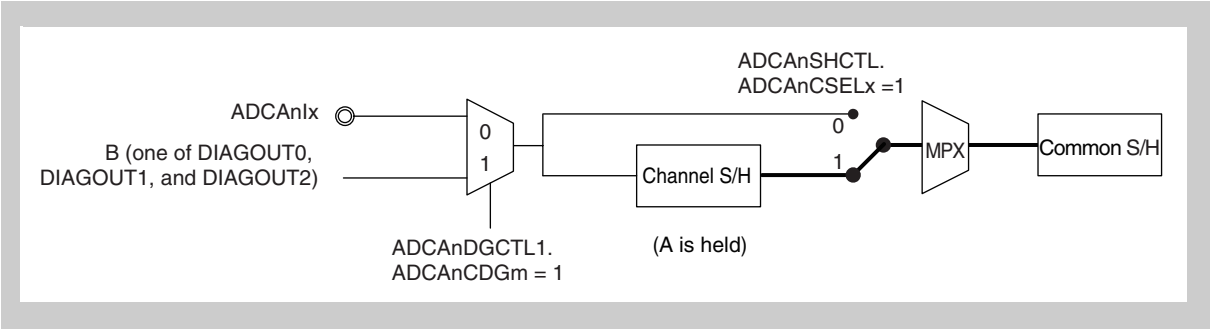
3. Set voltage "B" for DIAGOUT0, DIAGOUT1, or DIAGOUT2.

Conversion is performed without using the channel S/H circuit. The conversion result becomes "B".
The channel S/H circuit continues to hold "A".



4. Conversion is performed using the channel S/H circuit.

The channel S/H circuit continues to hold "A". The conversion result becomes “A”.



5. The result is as follows.
   – The value of the first conversion result (3) is “B”.
   – The value of the second conversion result (4) is “A”.

The diagnosis flow of the channel S/H circuit is shown below.

Initial settings

The following example uses the ADCAnI1 pin.

- Set ADCAnCTL1. ADCAnGPS to 1.
- Specify software trigger for the start trigger, and one-shot conversion mode (number of repetitions: 1) for the A/D conversion mode.
- Set the ADCAnCG0 register to 0000 0002_H. (ADCAnI1 with channel S/H circuit is selected.)
- Set the ADCAnIOC0 register to 0000 0002_H to generate the A/D conversion end interrupt INTADCAnT0. (An interrupt must be generated upon conversion end.)
- Set ADCAnDGCTL0. ADCAnPSEL[2:0] to 001_B. (1/3AV_{DD} is selected for reference voltage DIAGOUT1.)
- Set ADCAnDGCTL1. ADCAnCDG01 to 1.
- Set ADCAnCTL0. ADCAnCE to 1.

Operation flow

[Step 1]

Input the software start trigger for CG0 (first time). (1/3AV_{DD} is held in the channel S/H circuit.

Next, change the ADCAnPSEL[2:0] to 010_B. (Following completion of the first A/D conversion, 2/3AV_{DD} is selected for the DIAGOUT1 reference voltage.

Input the software start trigger for CG0 (second time). (The start trigger is maintained.)

The procedure up to this point must be performed until the first A/D conversion ends.

[Step 2]

The first A/D conversion ends and the A/D conversion end interrupt INTADCAnT0 is generated.

Change the ADCAnCSEL01 to 1. (The voltage held in the channel S/H circuit is converted at the next A/D conversion.)

Input the software start trigger for CG0 (third time). (The start trigger is maintained.)

The procedure up to this point must be performed until the second A/D conversion ends.

[Step 3]

The second A/D conversion ends and the A/D conversion end interrupt INTADCAnT0 is generated.

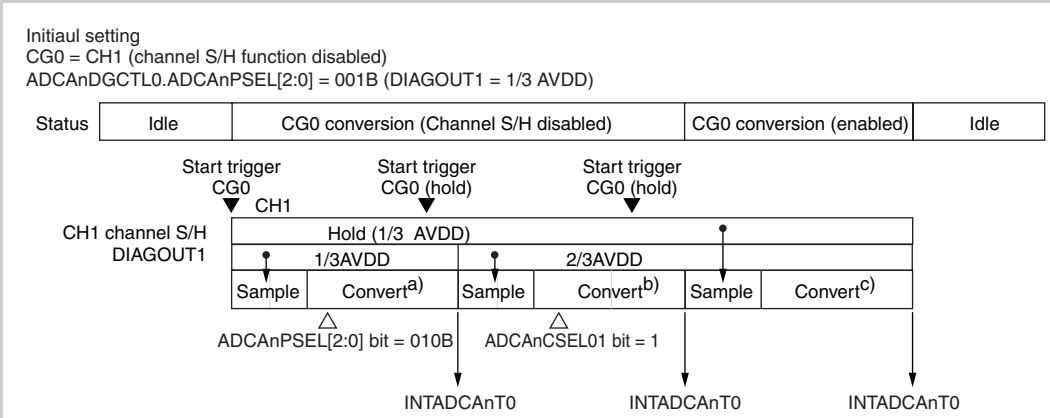
Read the ADCAnC01CR register and check whether it stores the expected value (2/3AV_{DD}).

[Step 4]

The third A/D conversion ends and the A/D conversion end interrupt INTADCAnT0 is generated.

Read the ADCAnC01CR register and check whether it stores the expected value (1/3AV_{DD}).

A timing example of the channel S/H circuit diagnosis is shown below.

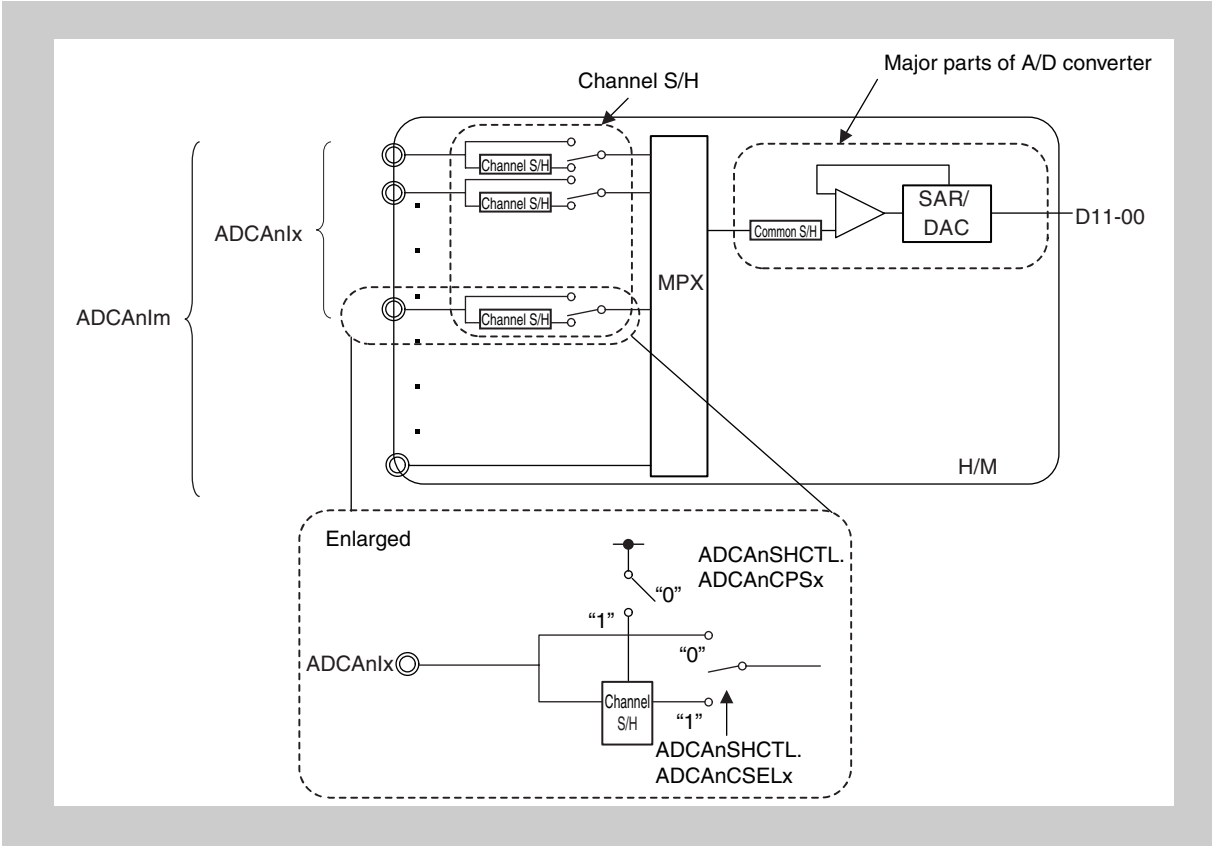


a) DIAGOUT1 is converted. The conversion result is 1/3 AV_{DD}.
b) DIAGOUT1 is converted. The conversion result is 2/3 AV_{DD}.
c) The input voltage held in channel S/H is converted. The conversion result is 1/3 AV_{DD}.

32.3.14 Channel S/H function (product dependent)

(1) Channel S/H function

The channel S/H circuit is as follows.



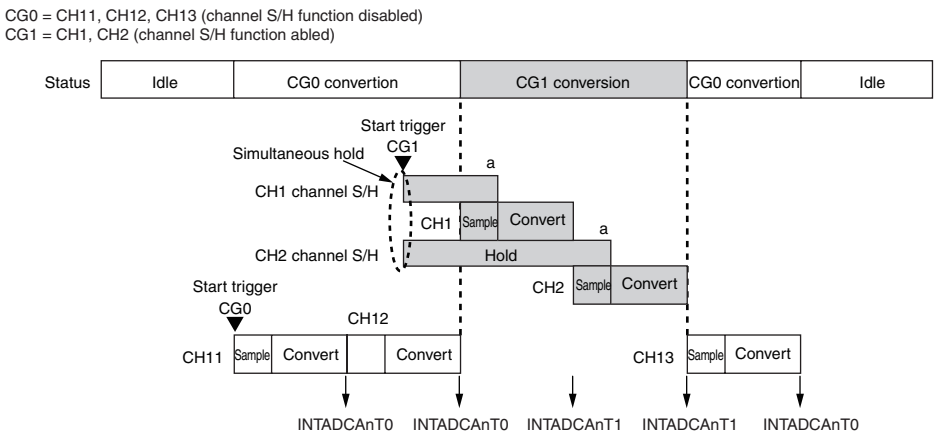
<R> **Note** When using the channel S/H function, enable the buffer amplifier function (by setting the ADCAnCTL1.ADCAnBPC bit to 1).

Channel S/H can be used in CG0's one-shot conversion mode (no repetition) and CG1, 2 (no repetition).

Power supply to the channel S/H circuit is controlled with ADCAnSHCTL. ADCAnCPSx, and the channel S/H function is enabled/disabled with ADCAnSHCTL. ADCAnCSELx.

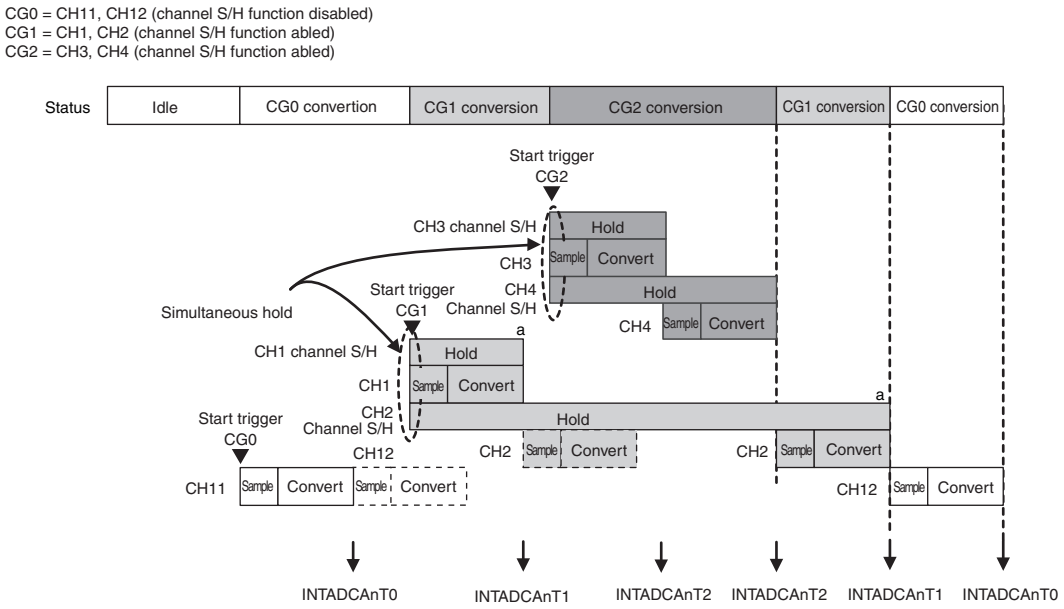
When a hardware start trigger or software start trigger is generated, the analog input signal of the channel for which the channel S/H function is enabled with ADCAnSHCTL.ADCAnCSELx is held with the channel S/H circuit. Next, scan list conversion starts according to the ADCAnCTL1.ADCAnTRMi setting.

Example 1: A start trigger with a high priority is generated while ADCAnTRM0 = 1 and ADCAnTRM1 = 1.



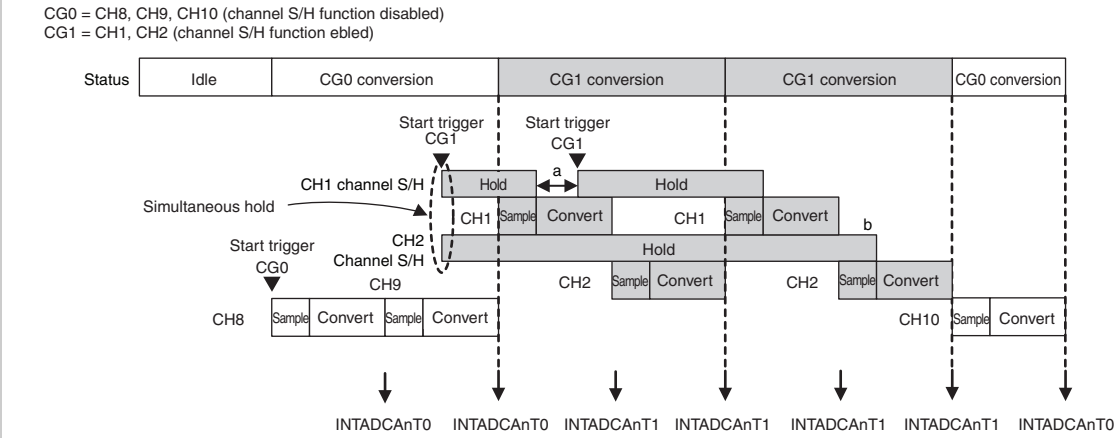
a) When ADCAnTRM1 bit = 1, the channel S/H circuit switches to sampling at this timing.

Example 2: A start trigger with a high priority is generated while ADCAnTRM0 = 0 and ADCAnTRM1 = 0.



a) When ADCAnTRM1 bit = 0, the channel S/H circuit switches to sampling at this timing.

Example 3: A subsequent start trigger is generated while ADCAnTRM0 = 1 and ADCAnTRM1 = 1.

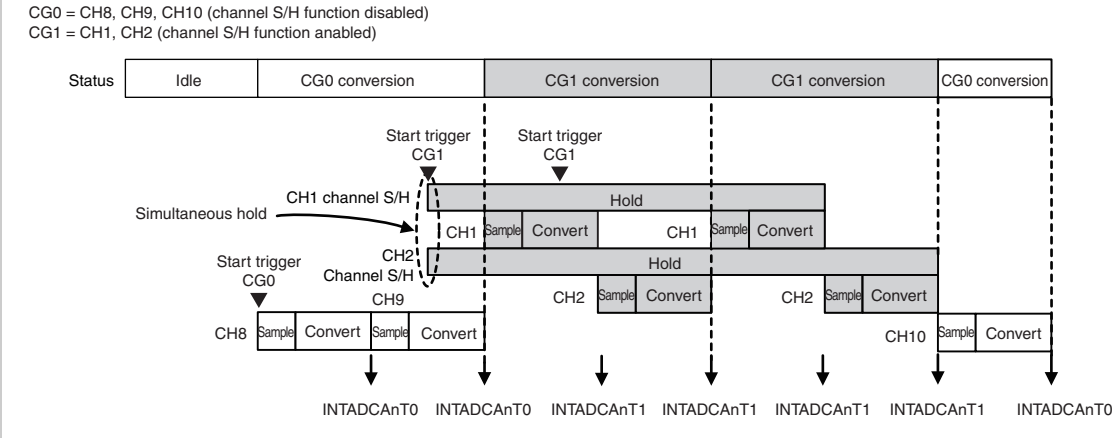


a) The sampling time in the channel S/H circuit must be taken into consideration. Secure 1.5 µs or more.

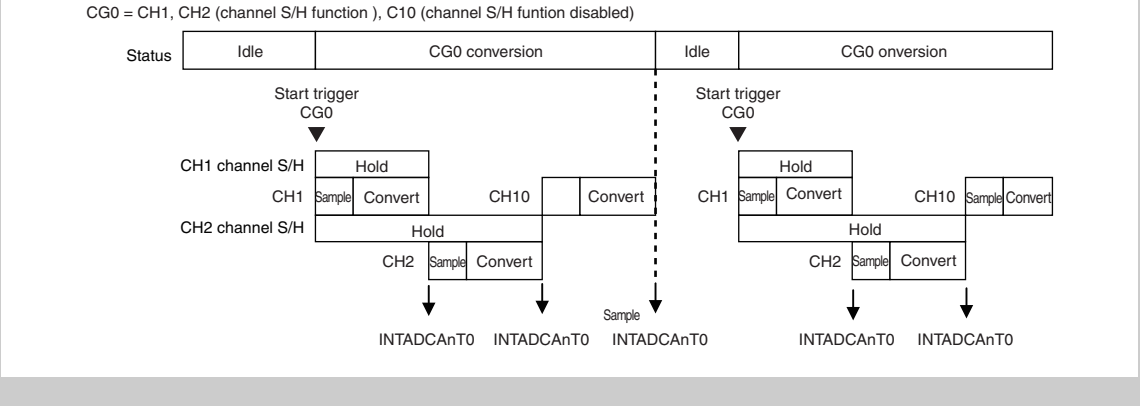
b) When the start trigger is generated, the channel S/H circuit continues to hold the current analog input until the next conversion.

Example 4: A subsequent start trigger is generated while ADCAnTRM0 = 1 and ADCAnTRM1 = 0.

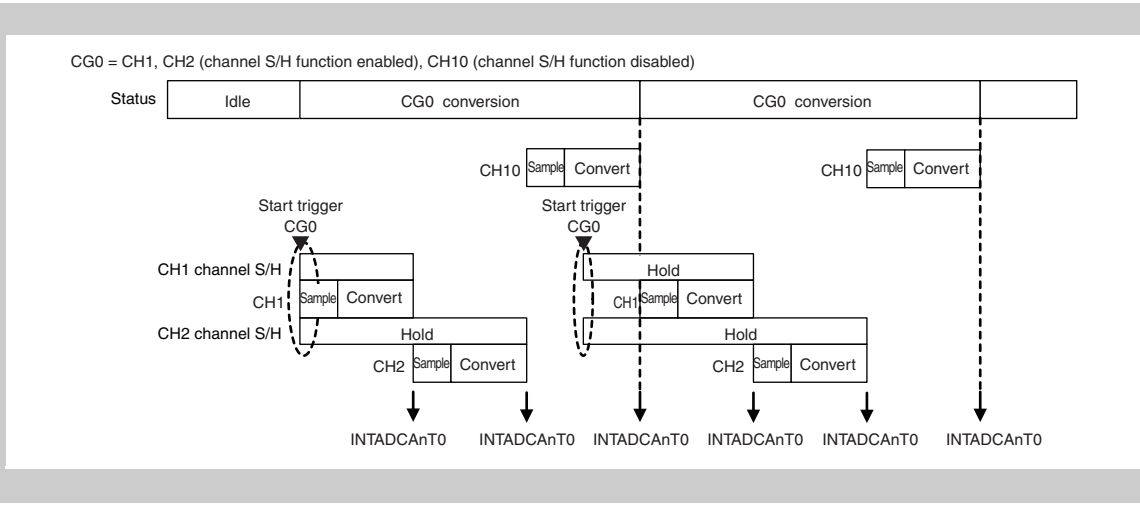
When a subsequent start trigger is generated during A/D conversion, the channel S/H circuit does not newly perform sampling.



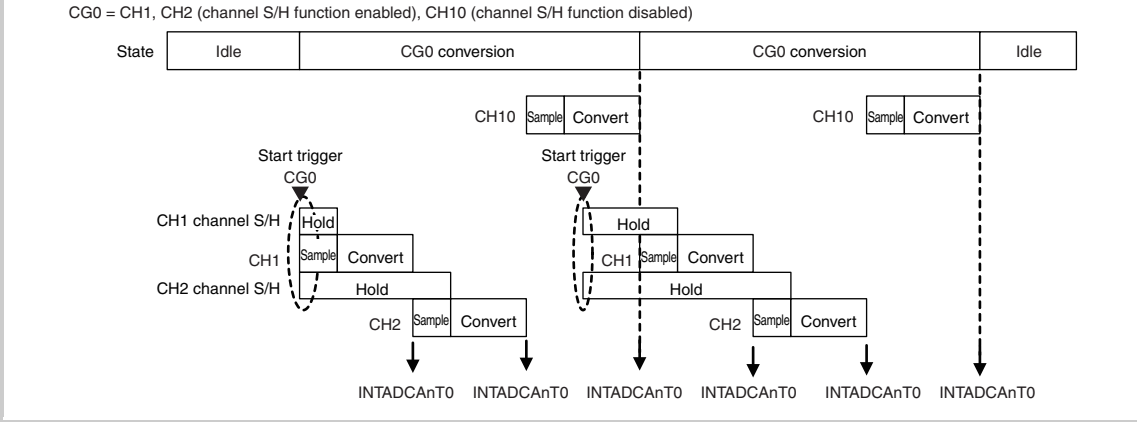
Example 5: CG0 is in one-shot conversion mode (no repetition) and ADCAnTRM0 = 0.



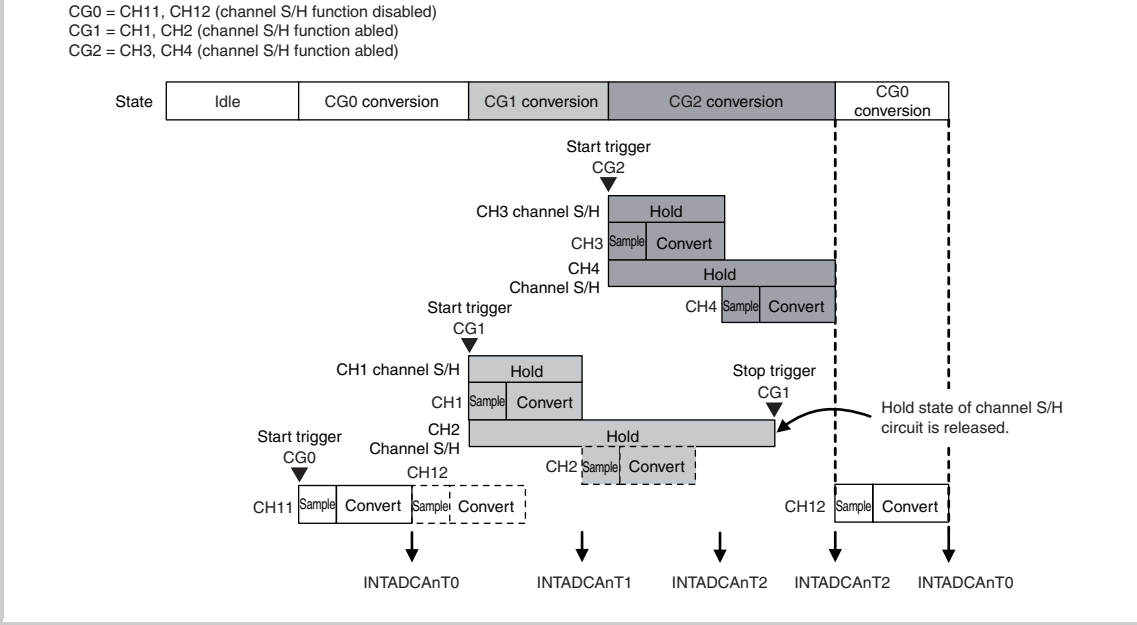
Example 6: A subsequent start trigger is generated while CG0 is in one-shot conversion mode (no repetition) and ADCAnTRM0 = 0.



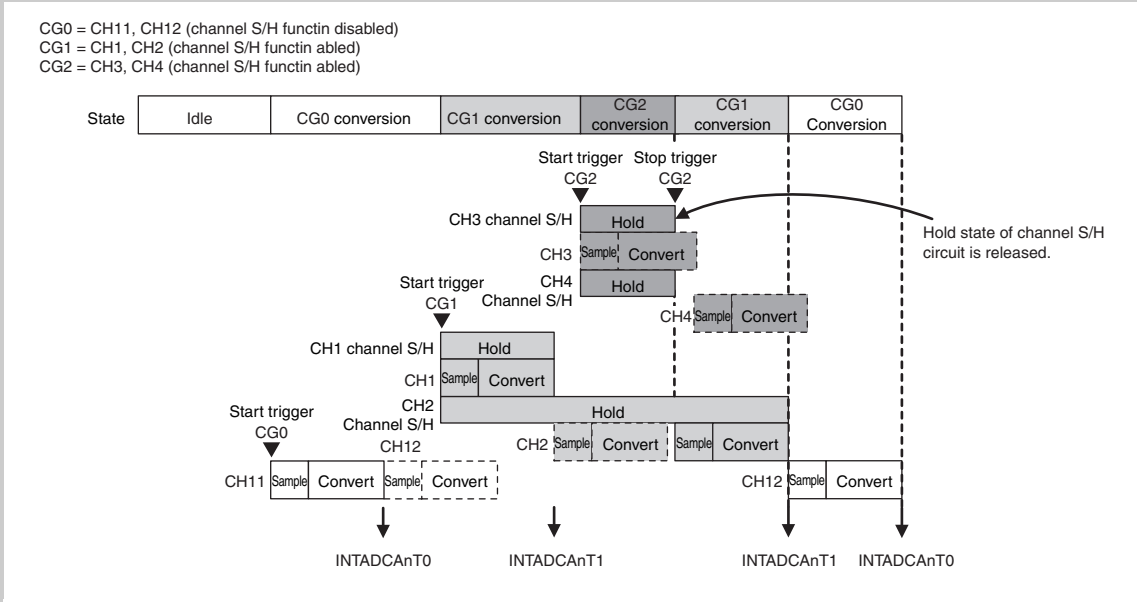
Example 7: A subsequent start trigger is generated while CG0 is in one-shot conversion mode (no repetition) and ADCAnTRM0 = 1.



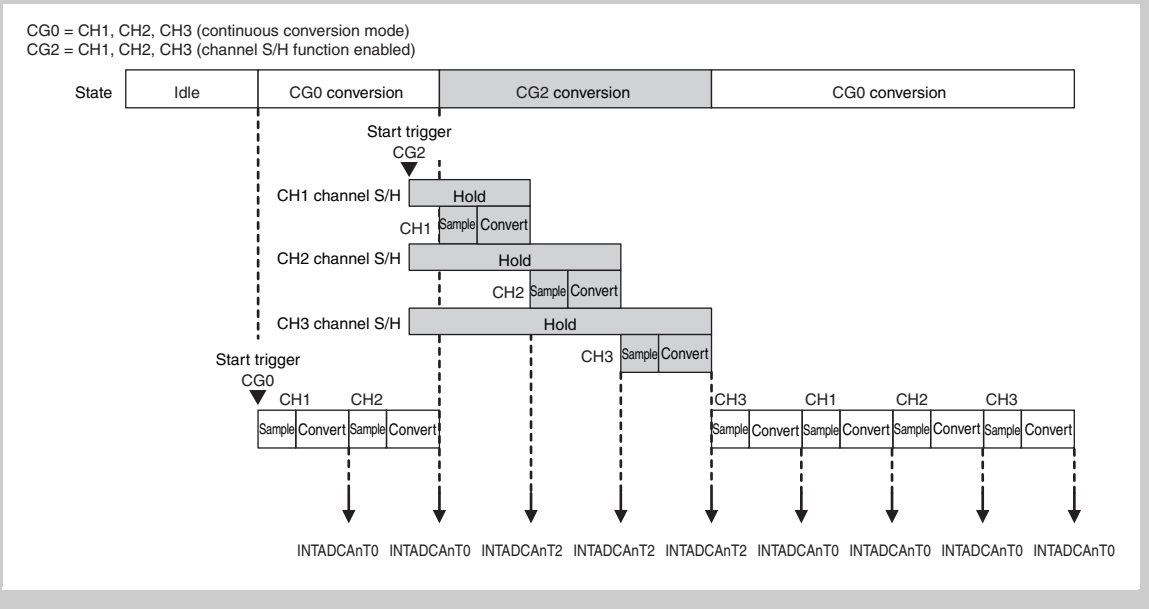
Example 8: The CG1 stop trigger is generated while different channels are set for CG1 and CG2, ADCAnTRM0 = 0, and ADCAnTRM1 = 0.



Example 9: The CG2 stop trigger is generated while different channels are set for CG1 and CG2, ADCAnTRM0 = 0, and ADCAnTRM1 = 0.



Example 10: The CG2 stop trigger is generated while different channels are set for CG1 and CG2, ADCAnTRM0 = 0, and ADCAnTRM1 = 0.



(2) Restrictions when using the channel S/H function

The restrictions that apply when the channel S/H function is used are described below.

1. Do not set a channel that will use the channel S/H function to multiple CGs.

Example 1:The channel S/H function is used for CG0, CG1, and CG2 (CG0 = one-shot conversion mode)

The following combinations can be set.

– CG0 (CH1 selected), CG1 (CH2 selected), CG2 (CH3 selected)

Setting the following combinations is prohibited.

– CG0 (CH1 selected), CG1 (CH1 selected), CG2 (CH2, 3 selected)

Example 2:The channel S/H function is used for CG1 and CG2 (CG0 = continuous conversion mode)

The following combinations can be set.

– CG0 (CH1 selected), CG1 (CH1, 2 selected), CG2 (CH3 selected)

– CG0 (CH1, 2, 3 selected), CG1 (CH1 selected), CG2 (CH2, 3 selected)

Setting the following combinations is prohibited.

– CG0 (CH1, 2, 3 selected), CG1 (CH1, 2 selected), CG2 (CH2, 3 selected)

2. To change a scan list during operation for a CGi that uses the channel S/H function, specify the settings so that the target channels that use the channel S/H function do not change.

Example: CH1, 2, and 3 use the channel S/H function

The following combinations can be set.

– CG0 (CH1, 2, 3) changed to CG0 (CH1, 2, 3, 10, 11)

– CG0 (CH1, 2, 3, 10, 11) changed to CG0 (CH1, 2, 3)

– CG0 (CH7, 8, 9) changed to CG0 (CH10, 11, 12)

Setting the following combinations is prohibited.

– CG0 (CH1, 2, 3) changed to CG0 (CH1, 2)

– CG0 (CH1) changed to CG0 (CH1, 2, 3)

– CG0 (CH7, 8, 9) changed to CG0 (CH1, 7, 8, 9)

– CG0 (CH1, 2, 9) changed to CG0 (CH9, 10, 11)

3. The repetition function is prohibited for a CG that uses the channel S/H function. If the channel S/H function is used in CGi, set the number of repetitions setting bits (ADCAnCTL0.ADCAnSCTi[1:0]) for that CG to 00_B.

32.3.15 Discharge function

If required, the internal capacitor of the common sample & hold circuit can be discharged prior to every conversion. This ensures that the capacitor is always empty before the new sample value is stored.

Note Using the discharge function increases the total conversion time by 1 ADCAnTCLK cycle (see *32.3.9 "Resolution, sampling and conversion times" on page 2347*).

Configuration The discharge function is enabled by setting ADCAnCTL1.ADCAnDISC to 1.

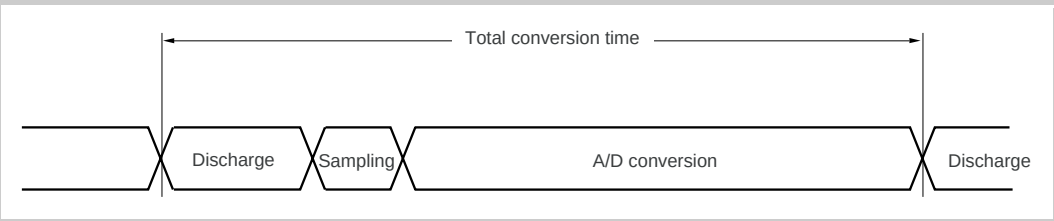


Figure 32-21 Timing when discharge function is enabled

32.3.16 Buffer amplifier function

If required, the analogue input signal can be connected to an internal buffer amplifier. The buffer amplifier accelerates charging of the internal sampling capacitor during the A/D sampling period.

The buffer amplifier function is enabled by setting ADCAnCTL1.ADCAnBPC to 1.

Note Using the buffer amplifier function increases the total conversion time by 4 ADCAnTCLK cycles (see *32.3.9 "Resolution, sampling and conversion times" on page 2347*).

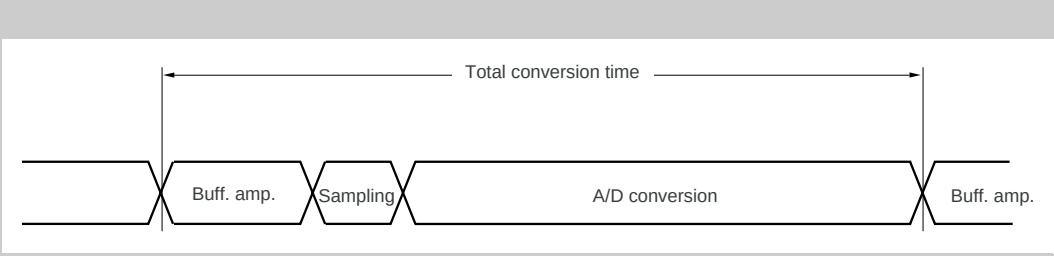


Figure 32-22 Timing when buffer amplifier function is enabled

32.3.17 Stabilization control

The A/D converter needs time for stabilization purposes in the following cases:

- The A/D converter has been switched on (ADCAnCTL1.ADCAnGPS = 1)
- The standby mode has been terminated.

A start trigger is accepted even while the stabilization time is being secured, but the conversion does not start before the stabilization time elapses.

In order to secure the minimum stabilization time, the stabilization time counter ADCAnCNT must be set (see the Data Sheet).

32.4 Registers

This section contains a description of all the registers of the ADCAn.

32.4.1 ADCAn register overview

The ADCAn is controlled and operated by the registers below.

- Where there is one register per channel, this is indicated by an “m”.
- Where there is one register per CG, this is indicated by an “i” (i = 0 to 2).

Table 32-13 ADCAn register overview (product dependent) (1/2)

Register name	Symbol	Address
Control registers		
A/D converter mode control register 0	ADCAnCTL0	<ADCAn_base0> + 100_H
A/D converter mode control register 1	ADCAnCTL1	<ADCAn_base0> + 104_H
A/D converter CG register i	ADCAnCGi	<ADCAn_base1> + i x 4_H
A/D converter interrupt control register i	ADCAnIOCi	<ADCAn_base1> + C_H + i x 4_H
A/D converter trigger select control register i	ADCAnTSELi	<ADCAn_base0> + 108_H + i x 4_H
A/D converter stabilization counter	ADCAnCNT	<ADCAn_base0> + 114_H
A/D converter channel S/H control register (product dependent)	ADCAnSHCTL	<ADCAn_base0> + 118_H
Conversion Status registers		
A/D converter overwrite error flag register	ADCAnSTR1	<ADCAn_base1> + 28_H
ADCAnSTR1 flag clear register	ADCAnSTC1	<ADCAn_base1> + 34_H
A/D converter status flag register 2	ADCAnSTR2	<ADCAn_base1> + $2C_H$
ADCAnSTR2 flag clear register	ADCAnSTC2	<ADCAn_base1> + 38_H
Software trigger registers		
A/D converter software trigger register i	ADCAnTRGi	<ADCAn_base1> + $A4_H$ + i x 4_H
A/D converter software trigger register 3	ADCAnTRG3	<ADCAn_base1> + $B0_H$
A/D converter software trigger register 4+i	ADCAnTRG4 + i	<ADCAn_base1> + $B4_H$ + i x 4_H
A/D converter software trigger register 7	ADCAnTRG7	<ADCAn_base1> + $C0_H$
A/D conversion result registers		
A/D converter latest conversion result register	ADCAnLCR	<ADCAn_base1> + $A0_H$
A/D converter conversion result register m	ADCAnCmCR	<ADCAn_base1> + $3C_H$ + m x 4_H
A/D converter CGi buffer result register i	ADCAnDBiCR	<ADCAn_base1> + $C4_H$ + i x 4_H
A/D converter diagnostic conversion result register	ADCAnDGCR	<ADCAn_base1> + $9C_H$
A/D conversion result upper/lower limit comparison registers		
A/D converter result check register	ADCAnCTL2	<ADCAn_base1> + 18_H
A/D converter result check (upper limit)	ADCAnUL	<ADCAn_base1> + $1C_H$
A/D converter result check (lower limit)	ADCAnLL	<ADCAn_base1> + 20_H
A/D converter result check error flag	ADCAnSTR0	<ADCAn_base1> + 24_H
ADCAnSTR0 flag clear register	ADCAnSTC0	<ADCAn_base1> + 30_H

Table 32-13 ADCAn register overview (product dependent) (2/2)

Register name	Symbol	Address
Diagnose functions control registers		
A/D converter self-diagnosis function control register 0	ADCAnDGCTL0	<ADCAn_base1> + DC_H
A/D converter self-diagnosis function control register 1	ADCAnDGCTL1	<ADCAn_base0> + $11C_H$
A/D converter internal pull down resistance control register 0	ADCAnPDCTL0	<ADCAn_base0> + 120_H
A/D converter channel S/H control register (product dependent)	ADCAnSHCTL	<ADCAn_base0> + 118_H

32.4.2 Control registers

(1) ADCAnCTL0 – A/D converter mode control register 0

This register enables/disables the A/D converter. Additionally, it specifies the number of repetitions for one-shot conversion mode, and also specifies whether to generate an error interrupt request if the A/D conversion result is overwritten before it is read.

Access This register can be read/written in16-bit units.

Address <ADCAn_base0> + 100_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	ADCAn OEM4	ADCAn OEM[3:1]			ADCAn OEM0	ADCAn CE	0	ADCAn SCT2[1:0]		ADCAn SCT1[1:0]		ADCAn SCT0[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 32-14 ADCAnCTL0 register contents (1/2)

Bit position	Bit name	Function
12	ADCAn OEM4	Specifies whether to generate the INTADCAnTERR error interrupt if this bit is overwritten before the A/D conversion result of the ADCAnLCR register is read. 0: Generate the INTADCAnTERR error interrupt if the A/D conversion result is overwritten. 1: Do not generate the INTADCAnTERR error interrupt. See *(1) "Conversion result overwrite check function" on page 2353* for details.
11 to 9	ADCAn OEM[3:1]	Specifies whether to generate the INTADCAnTERR error interrupt if the A/D conversion result in any ADCAnDBiCR register is overwritten before it is read. 0: Generate the INTADCAnTERR error interrupt if the A/D conversion result is overwritten. 1: Do not generate the INTADCAnTERR error interrupt. CGi is controlled with the ADCAnOEM(i+1) bit. See *(1) "Conversion result overwrite check function" on page 2353* for details.
8	ADCAn OEM0	Specifies whether to generate the INTADCAnTERR error interrupt if the A/D conversion result in the ADCAnCmCR register is overwritten before it is read. 0: Generate the INTADCAnTERR error interrupt if the A/D conversion result is overwritten. 1: Do not generate the INTADCAnTERR error interrupt. See *(1) "Conversion result overwrite check function" on page 2353* for details.
7	ADCAn CE	This bit enables/disables the A/D converter. 0: Disable A/D converter 1: Enable A/D converter Note that A/D conversion only starts on hardware trigger or software trigger (ADCAnTRGi.ADCAnSTTi) – when ADCAnCTL0.ADCAnCE = 1. Note also that the A/D converter needs time to stabilize after it has been enabled. A start trigger is accepted even immediately after power-on. After the values of the stabilization counter ADCAnCNT changes to 00_H, the A/D conversion starts.

Table 32-14 ADCAnCTL0 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>5 to 0</td><td>ADCAn SCTi[1:0]</td><td>Number of repetitions of scan list conversion for CG1, CG2, and CG0 in one-shot conversion mode
<table>
<tr><th>ADCAn SCTi1</th><th>ADCAn SCTi0</th><th>Number of repetitions of scan list conversion for CGi</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>0</td><td>3</td></tr>
<tr><td>1</td><td>1</td><td>4</td></tr>
</table></td></tr>
</table>

(2) ADCAnCTL1 – A/D converter mode control register 1

This register specifies the conversion mode and controls the conversion operations.

Access This register can be read/written in 32-bit units.

Address <ADCAn_base0> + 104_H

Initial Value $0100\ 0008_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAn T2ETS[1:0]		ADCAn T1ETS[1:0]		ADCAn T0ETS[1:0]		0	ADCAn CRAC	0	0	ADCAn MD1	ADCAn MD0	0	0	ADCAn DISC	ADCAn RCL
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R	R	R/W	R/W	R	R	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAn CTYP	0	0	ADCAn STL	ADCAn FR[3:0]				0	ADCAn TRM[2:0]			ADCAn BPC	0	0	ADCAn GPS
R/W	R	R	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R	R	R/W

Table 32-15 ADCAnCTL1 register contents (1/2)

Bit position	Bit name	Function
31 to 26	ADCAn TiETS[1:0]	Specifies the valid edge of the hardware trigger ADCATTRGi signal. **ADCAnTiETS1 / ADCAnTiETS0 / Valid edge** 0 / 0 / No edge detection (not accepted) 0 / 1 / Rising edge 1 / 0 / Setting prohibited 1 / 1 / Setting prohibited
24	ADCAn CRAC	Specifies the alignment of the A/D conversion results and diagnosis conversion results. 0: Right-aligned 1: Left-aligned
21	ADCAn MD1	Specifies the A/D conversion start trigger for all CGs. 0: Software trigger 1: Hardware trigger and software trigger This configuration is valid for all CGs. The A/D converter only detects triggers when the A/D converter is enabled. See *32.3.5 "Starting A/D conversion (start trigger)" on page 2342* for details.
20	ADCAn MD0	Specifies the A/D conversion mode for CG0. 0: One-shot conversion mode The iteration factor is configured in ADCAnCTL0.ADCAnSCTi[1:0] for each CG individually. 1: Continuous conversion mode This configuration applies to the A/D conversion of CG0 only. CG1 and CG2 always operates in one-shot conversion mode. See *32.3.4 "A/D conversion modes" on page 2340* for details.
17	ADCAn DISC	Enables/disables the discharge function. 0: Disable 1: Enable See *32.3.15 "Discharge function" on page 2369* for details.

Table 32-15 ADCAnCTL1 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>16</td><td>ADCAnRCL</td><td>Specifies whether the A/D conversion results ADCAnCmCR and ADCAnDBiCR are retained after they are read.
0: A/D conversion results are retained until they are overwritten by the next A/D conversion results.
1: A/D conversion results are cleared after they are read.</td></tr>
<tr><td>15</td><td>ADCAnCTYP</td><td>Specifies the resolution mode.
0: 12-bit resolution (product-dependent)
1: 10-bit resolution</td></tr>
<tr><td>12</td><td>ADCAnSTL</td><td>Specifies the level of the ADCAnCNVi signal.
0: When ADCAnCNVi = L, CGi conversion is not in progress.
When ADCAnCNVi = H, CGi conversion is in progress.
1: When ADCAnCNVi = H, CGi conversion is not in progress.
When ADCAnCNVi = L, CGi conversion is in progress.</td></tr>
<tr><td>11 to 8</td><td>ADCAn FR[3:0]</td><td>Specifies the ADCAn clock ADCAnTCLK
<table>
<tr><th>ADCAnFR[3:0]</th><th>ADCAn clock</th></tr>
<tr><td>0000</td><td>PCLK / 2</td></tr>
<tr><td>0001</td><td>PCLK / 3</td></tr>
<tr><td>0010</td><td>PCLK / 4</td></tr>
<tr><td>0011</td><td>PCLK / 5</td></tr>
<tr><td>0100</td><td>PCLK / 6</td></tr>
<tr><td>0110</td><td>PCLK / 8</td></tr>
<tr><td>1000</td><td>PCLK / 10</td></tr>
<tr><td>1010</td><td>PCLK / 12</td></tr>
<tr><td>1100</td><td>PCLK / 14</td></tr>
<tr><td>1110</td><td>PCLK / 16</td></tr>
<tr><td>all others</td><td>setting prohibited</td></tr>
</table></td></tr>
<tr><td>6 to 4</td><td>ADCAnTRMi (product dependent)</td><td>Specifies the halt operation when the start trigger of the A/D conversion for a CG with higher priority is input (or when transition to ADCHALT mode is requested).
0: Immediately halt the current A/D conversion for CGi and start A/D conversion for the CG with the higher priority (or enter ADCHALT mode).
1: Halt the A/D conversion for CG after the current A/D conversion for CGi ends, and then start the A/D conversion for the CG with the higher priority (or enter ADCHALT mode).
Following the end of the A/D conversion for the CG with the higher priority (or return from ADCHALT mode), the A/D conversion for CGi is resumed.
The prioritization is as follows: ADCHALT > CG2 > CG1 > CG0
See (1) “Order of A/D conversion” on page 2338 for details.</td></tr>
<tr><td>3</td><td>ADCAnBPC</td><td>Enables/disables the buffer amplifier function:
0: Disable
1: Enable
See 32.3.16 “Buffer amplifier function” on page 2369 for details.</td></tr>
<tr><td>0</td><td>ADCAn GPS</td><td>Switches the power of the ADCAn off/on:
0: Power off
1: Power on
The A/D converter needs a stabilization time after A/D power on (see 32.3.17 “Stabilization control” on page 2370).</td></tr>
</table>

(3) ADCAnCGi – A/D converter channel group register i

This register creates the scan list for each CG. The channels set to the scan list are converted in ascending order, starting from the lowest channel number. For details, see *32.3.3 "Channels and channel groups" on page 2338*.

It is also possible to enable/disable diagnosis of A/D conversion using the ADDIAGOUT reference voltage signal, with ADCAnCG0.ADCAnDIAG. For details, see *(1) "Diagnosis for A/D conversion circuit" on page 2356*.

Access This register can be read/written in 32-bit units.
This register is a master/slave configuration register, and it can specify a new A/D conversion channel to the master register during A/D conversion. The timing at which the value of the master register is transferred to the slave register is as follows.

- When CGi is not currently undergoing A/D conversion, the value of the master register is transferred to the slave register one clock (PCLK) after it is written to the master register.
- When CGi is currently undergoing A/D conversion, the value of the master register is transferred to the slave register upon completion of the scan list conversion of CGi currently being executed.
- When the stop trigger bit (ADCAnSPi) of CGi is set following write to this register, the value of the master register is transferred to the slave register when A/D conversion is stopped.

Address <ADCAn_base1> + i x 4_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAn DIAG	0	0	0	0	0	0	0	ADCAnCGiS[23:16]							
R/W	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCGiS[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 32-16 ADCAnCGi register contents

Bit position	Bit name	Function
31	ADCAn DIAG	Enables/disables the diagnostic A/D conversion of the reference voltage ADDIAGOUT signal at the end of the A/D conversion of CG0: 0: Disable A/D conversion of ADDIAGOUT signal 1: Convert ADDIAGOUT signal This bit can be set only for ADCAnCG0. For ADCAnCG1 and ADCAnCG2, set this bit to 0.
<R> 23 to 00	ADCAn CGiS[23:00]	Specifies the analog inputs to be converted for CGi: 0: Do not convert analog input ADCAnIm 1: Convert analog input ADCAnIm Note: Set the bits corresponding to channels not provided in this product to 0.

(4) ADCAnIOCi – A/D converter interrupt control register i

The A/D conversion end interrupt INTADCAnTi can be generated when the A/D conversion of a certain channel has been completed.

This register specifies the channels for which INTADCAnTi is generated on A/D conversion completion.

If ADCAnIOCi = 0000 0000_H, INTADCAnTi is automatically generated at the completion of A/D conversion of CGi.

Access This register can be read/written in 32-bit units.
It can be written at any time even when the A/D converter is enabled (ADCAnCTL0.ADCAnCE = 1). The new value takes effect after the current A/D conversion of CGi has been completed.

Address <ADCAn_base1> + $0C_H$ + i x 4_H

Initial Value 0000 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAnCG0IDG	0	0	0	0	0	0	0	ADCAnCGil[23:16]							
R/W	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCGil[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 32-17 ADCAnIOCi register contents

Bit position	Bit name	Function
31	ADCAn CG0IDG	Specifies whether INTADCAnTi is generated on completion of the A/D conversion of the reference voltage when the diagnostic mode is enabled for CG0 (ADCAnCG0.ADCAnDIAG = 1): 0: Do not generate INTADCAnTi 1: Generate INTADCAnTi This bit can be set only in the ADCAnIOC0 register. Set this bit in the ADCAnIOC1 register and the ADCAnIOC2 register to 0. See *(1) "Diagnosis for A/D conversion circuit" on page 2356* for details.
23 to 00	ADCAn CGil[23:00]	Specifies whether or not the interrupt INTADCAnTi is generated on A/D conversion completion of channel m: 0: Do not generate INTADCAnTi 1: Generate INTADCAnTi Note: Set the bits corresponding to channels not provided in this product to 0.

Note As the ADCAnIOCi register is associated with the ADCAnCGi register, their buffer registers must be updated simultaneously. As the update time depends on writing ADCAnCGi, always write ADCAnIOCi before ADCAnCGi if you want to change the interrupt generation for a CG.

(5) ADCAnCNT – A/D converter stabilization counter

This register specifies the stabilization time.

Access This register can be read/written in 8-bit units.

Address <ADCAn_base0> + 114_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
ADCAnCNT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 32-18 ADCAnCNT register contents

Bit position	Bit name	Function
7 to 0	ADCAnCNT[7:0]	Specifies the stabilization counter: Stabilization time = ADCAnCNT[7:9] × clock cycles (PCLK)

(6) ADCAnTSELi – A/D converter trigger select control register 0 (product-dependent)

This register specifies the input signals to be used in combination with hardware start trigger ADCAnTTRGi signals. Multiple trigger sources can be used simultaneously.

Access This register can be read/written in 16-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0)

Address <ADCAn_base0> + 108_H + i x 4_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnTiSEL[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 32-19 ADCAnTSELi register contents

Bit position	Bit name	Function
15 to 0	ADCAnTiSEL[15:00]	Specifies whether the corresponding input signal is to be used as hardware start trigger 0: Do not use as hardware start trigger 1: Use as hardware start trigger Note: Set the bits corresponding to channels not provided in this product to 0.

Note For the connection destination of the hardware start trigger, see “Connection destination of hardware trigger” in the first section of this chapter.

32.4.3 Conversion status registers

(1) ADCAnSTR1 – A/D converter overwrite error flag

This register indicates whether the latest A/D conversion result is overwritten before it is read.

Access This register can be read in 32-bit units.

Address <ADCAn_base1> + 28_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnOWE[23:16]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnOWE[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 32-20 ADCAnSTR1 register contents

Bit position	Bit name	Function
23 to 0	ADCAn OWE[23:00]	Indicates whether the A/D conversion result of channel m is overwritten before it is read. 0: Not overwritten 1: Overwritten This error flag is cleared by setting ADCAnSTC1.ADCAnOWECm to 1. Note: Set the bits corresponding to channels not provided in this product to 0.

Note ADCAnSTR1.ADCAnOWEm is mirrored by the following overwrite error flag:

- Error flag in A/D converter conversion result register for channel m (ADCAnCmCR.ADCAnCmER1)

(2) ADCAnSTC1 – ADCAnSTR1 flag clear register

This register is the clear control register of ADCAnSTR1.

Access This register can be written in 32-bit units.

It is always read as 0000 $0000_{H.}$

Address <ADCAn_base1> + 34_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnWEC[23:16]							
R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnWEC[15:00]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 32-21 ADCAnSTC1 register contents

Bit position	Bit name	Function
23 to 0	ADCAn OWEC[23:00]	0: No function 1: Clears the corresponding ADCAnSTR1.ADCAnOWEm Note: Set the bits corresponding to channels not provided in this product to 0.

(3) ADCAnSTR2 – A/D converter status flag 2

This register indicates the current conversion status.

Access This register can be read in 16-bit units.

Address <ADCAn_base1> + $2C_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	ADCAnRQT3	ADCAnRQ[2:0]			0	0	0	0	ADCAnST3	ADCAnST[2:0]		
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 32-22 ADCAnSTR2 register contents

Bit position	Bit name	Function
11	ADCAnRQ3 (product-dependent)	Indicates whether a ADCHALT request is pending: 0: ADCHALT request is not pending 1: ADCHALT request is pending
10 to 8	ADCAnRQ[2:0]	Indicates whether A/D conversion request for CGi is pending: 0: A/D conversion request for CGi is not pending 1: A/D conversion request for CGi is pending
3	ADCAnST3 (product-dependent)	Indicates whether the A/D conversion is currently in the ADCHALT state due to a software trigger (ADCAnTRG3.ADCAnSTT3). 0: Not in ADCHALT state 1: In ADCHALT state This bit is cleared when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).
2 to 0	ADCAnST[2:0]	Indicates whether A/D conversion of CGi is currently performed: 0: A/D conversion is not currently performed (including halt caused by A/D conversion of CG with higher priority) 1: A/D conversion is currently performed This bit is cleared when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

(4) ADCAnSTC2 – A/D converter status flag clear register 2

This register is used to clear the overwrite and result check status flags of ADCAnLCR and ADCAnDBiCR.

Access This register can be written in 8-bit units.

Address <ADCAn_base1> + 38_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
ADCAn LERC1	ADCAn LERC0	ADCAn DB2ERC1	ADCAn DB2ERC0	ADCAn DB1ERC1	ADCAn DB1ERC0	ADCAn DB0ERC1	ADCAn DB0ERC0
W	W	W	W	W	W	W	W

Table 32-23 ADCAnSTC2 register contents

Bit position	Bit name	Function
7	ADCAn LERC1	Clears the overwrite flag ADCAnLCR.ADCAnLER1: 0: No function 1: ADCAnLCR.ADCAnLER1 is cleared
6	ADCAn LERC0	Clears the result check error flag ADCAnLCR.ADCAnLER0: 0: No function 1: ADCAnLCR.ADCAnLER0 is cleared
5, 3, 1	ADCAn DBiERC1	Clears the overwrite flag ADCAnDBiCR.ADCAnDBiER1: 0: No function 1: ADCAnDBiCR.ADCAnDBiER1 is cleared
4, 2, 0	ADCAn DBiERC0	Clears the result check error flag ADCAnDBiCR.ADCAnDBiER0: 0: No function 1: ADCAnDBiCR.ADCAnDBiER0 is cleared

32.4.4 Software trigger registers

(1) ADCAnTRGi – A/D converter software trigger register i

This register is the trigger register to *start* the A/D conversion of CGi.

Access This register can be written in 8-bit units.
It is always read as $00_{H.}$

Address <ADCAn_base1> + $A4_H$ + i x 4_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ADCAn STTi
R	R	R	R	R	R	R	W

Table 32-24 ADCAnTRGi register contents

Bit position	Bit name	Function
0	ADCAnSTTi	Starts the A/D conversion of CGi: 0: No function 1: Starts A/D conversion of CGi

See *32.3.5 “Starting A/D conversion (start trigger)” on page 2342* for details.

(2) ADCAnTRG3 – A/D converter software trigger register 3 (product-dependent)

This register is the trigger register for transition to ADCHALT mode.

Access This register can be written in 8-bit units.
It is always read as $00_{H.}$

Address <ADCAn_base1> + $B0_H$

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ADCAn STT3
R	R	R	R	R	R	R	W

Table 32-25 ADCAnTRG3 register contents

Bit position	Bit name	Function
0	ADCAnSTT3	0: No function 1: Transition to ADCHALT mode

See *32.3.8 “Pausing and resuming A/D conversion (ADCHALT mode) (product dependent)” on page 2346* for details.

(3) ADCAnTRGi+4 – A/D converter software trigger register i+4

This register is the software trigger register to *stop* the A/D conversion of CGi.

Access This register can be written in 8-bit units.
It is always read as $00_{H.}$

Address <ADCAn_base1> + $B4_H$ + i x 4_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ADCAn SPi
R	R	R	R	R	R	R	W

Table 32-26 ADCAnTRG4 + i register contents

Bit position	Bit name	Function
0	ADCAnSPi	0: No function 1: Stops the A/D conversion of CGi

See *32.3.6 “Stopping A/D conversion (stop trigger)” on page 2344* for details.

(4) ADCAnTRG7 – A/D converter software trigger register 7 (product-dependent)

This register is the software trigger register to resume the A/D conversion after cancelling ADCHALT mode.

Access This register can be written in 8-bit units.
It is always read as $00_{H.}$

Address <ADCAn_base1> + $C0_H$

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ADCAn SP3
R	R	R	R	R	R	R	W

Table 32-27 ADCAnTRG7 register contents

Bit position	Bit name	Function
0	ADCAnSP3	0: No function 1: Resume A/D conversion

See *32.3.8 “Pausing and resuming A/D conversion (ADCHALT mode) (product dependent)” on page 2346* for details.

32.4.5 A/D conversion result registers

(1) ADCAnLCR – A/D converter latest conversion result register

This register stores the result and the status of the latest A/D conversion.

It allows you to read the latest A/D conversion result.

Access This register can be read in 32-bit units.

- The upper 16 bits store the A/D conversion result status.
- The lower 16 bits store the A/D conversion result.

Address <ADCAn_base1> + $A0_H$

Initial Value $0300\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	ADCAn LCG[1:0]		ADCAn LER1	ADCAn LER0	ADCAn LUR	ADCAnLCN[4:0]				
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnLCR[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 32-28 ADCAnLCR register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>25, 24</td><td>ADCAn LCG[1:0]</td><td>Specifies the CG to which the result in ADCAnLCR[15:00] belongs.
<table>
<tr><th>ADCAn LCG1</th><th>ADCAn LCG0</th><th>Channel group</th></tr>
<tr><td>0</td><td>0</td><td>CG0</td></tr>
<tr><td>0</td><td>1</td><td>CG1</td></tr>
<tr><td>1</td><td>0</td><td>CG2</td></tr>
<tr><td>1</td><td>1</td><td>None</td></tr>
</table></td></tr>
<tr><td>23</td><td>ADCAnLER1</td><td>Indicates the overwrite error status.
0: Not overwritten
1: Overwritten
This error flag is cleared when ADCAnSTC2.ADCAnLERC1 is set to 1.</td></tr>
<tr><td>22</td><td>ADCAnLER0</td><td>Indicates the A/D conversion result upper/lower limit comparison status.
0: The conversion result is within the setting range.
1: The conversion result is out of the setting range.
This error flag is cleared when ADCAnSTC2.ADCAnLERC0 is set to 1.</td></tr>
<tr><td>21</td><td>ADCAnLUR</td><td>Indicates the update status of the A/D conversion result.
0: The A/D conversion result is read from the ADCAnLCR register.
1: This is the latest value, and the A/D conversion result is not read from the ADCAnLCR register.
This bit is cleared after the read operation.</td></tr>
<tr><td>20 to 16</td><td>ADCAn LCN[4:0]</td><td>Indicates the channel number corresponding to the conversion result stored in the ADCAnLCR[15:00].
00001 × m = CHm</td></tr>
<tr><td>15 to 0</td><td>ADCAn LCR[15:00]</td><td>Indicates the result of the A/D conversion.
The resolution and alignment are determined by ADCAnCTL1.ADCAnCTYP and ADCAnCTL1.ADCAnCRAC, as follows.
<table>
<tr><th>ADCAn CTL1. ADCAn CTYP</th><th>ADCAn CTL1. ADCAn CRAC</th><th>Resolution and alignment</th><th>Bit positions of A/D conversion result</th></tr>
<tr><td>0</td><td>0</td><td>12-bit resolution, right-aligned</td><td>[11:00] of ADCAnLCR[15:00]</td></tr>
<tr><td>0</td><td>1</td><td>12-bit resolution, left-aligned</td><td>[15:04] of ADCAnLCR[15:00]</td></tr>
<tr><td>1</td><td>0</td><td>10-bit resolution, right-aligned</td><td>[09:00] of ADCAnLCR[15:00]</td></tr>
<tr><td>1</td><td>1</td><td>10-bit resolution, left-aligned</td><td>[15:06] of ADCAnLCR[15:00]</td></tr>
</table></td></tr>
</table>

Note If A/D conversion is performed using the internal reference voltage, the A/D conversion result is stored in ADCAnDGCR, not in the ADCAnLCR, ADCAnCmCR, and ADCAnDBiCR (see *(4) "ADCAnDGCR – Diagnostic conversion result register" on page 2392*).

(2) ADCAnCmCR – A/D converter conversion result register for channel m

This register stores the result and the status of the latest A/D conversion of channel m.

It allows you to read the A/D conversion results for a channel.

Access This register can be read in 32-bit units.

- The upper 16 bits store the A/D conversion result status.
- The lower 16 bits store the A/D conversion result.

Address <ADCAn_base1> + $3C_H$ + m x 4_H

Initial Value $0300\ 0000_H$ + m x $0001\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	ADCAn CmCG[1:0]		ADCAn CmER1	ADCAn CmER0	ADCAn CmUR	ADCAnCmCN[4:0]				
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCmCR[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Notes

1. The functions of the individual bits are identical to those of the ADCAnLCR register, except that this register indicates the latest A/D conversion result for a specific CGi rather than for all the CGs like the ADCAnLCR register (see *Table 32-28 "ADCAnLCR register contents" on page 2387*).
2. After reset, ADCAnCmCG[1:0] = 11_B.
3. If ADCAnCTL1.ADCAnRCL is set to 0, the A/D conversion result in ADCAnCmCR[15:00] is kept until it is overwritten by the next A/D conversion result.

   If ADCAnCTL1.ADCAnRCL is set to 1, ADCAnCmCR[15:00] is cleared by reading it.

Table 32-29 ADCAnCmCR register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>25, 24</td><td>ADCAn CmCG[1:0]</td><td>Specifies the CG to which the result in ADCAnCmCR[15:00] belongs.
<table>
<tr><th>ADCAn CmCG1</th><th>ADCAn CmCG0</th><th>Channel group</th></tr>
<tr><td>0</td><td>0</td><td>CG0</td></tr>
<tr><td>0</td><td>1</td><td>CG1</td></tr>
<tr><td>1</td><td>0</td><td>CG2</td></tr>
<tr><td>1</td><td>1</td><td>None</td></tr>
</table></td></tr>
<tr><td>23</td><td>ADCAn CmER1</td><td>Indicates the overwrite error status.
0: Not overwritten
1: Overwritten
This error flag reflects the value of ADCAnSTR1.ADCAnOWEm. It is cleared when ADCAnSTC1.ADCAnQWECm is set to 1.</td></tr>
<tr><td>22</td><td>ADCAn CmER0</td><td>Indicates the A/D conversion result upper/lower limit comparison status.
0: The conversion result is within the setting range.
1: The conversion result is out of the setting range.
This error flag reflects the value of ADCAnSTR0.ADCAnRCEm. It is cleared when ADCAnSTC0.ADCAnRCECm is set to 1.</td></tr>
<tr><td>21</td><td>ADCAn CmUR</td><td>Indicates the update status of the A/D conversion result.
0: The A/D conversion result is read from the ADCAnCmCR register.
1: This is the latest value, and the A/D conversion result is not read from the ADCAnCmCR register.
This bit is cleared after the read operation.</td></tr>
<tr><td>20 to 16</td><td>ADCAn CmCN[4:0]</td><td>Indicates the channel number corresponding to the conversion result stored in the ADCAnCmCR[15:00].
00001 × m = CHm</td></tr>
<tr><td>15 to 0</td><td>ADCAn CmCR[15:00]</td><td>Indicates the result of the A/D conversion.
The resolution and alignment are determined by ADCAnCTL1.ADCAnCTYP and ADCAnCTL1.ADCAnCRAC, as follows.
<table>
<tr><th>ADCAn CTL1. ADCAn CTYP</th><th>ADCAn CTL1. ADCAn CRAC</th><th>Resolution and alignment</th><th>Bit positions of A/D conversion result</th></tr>
<tr><td>0</td><td>0</td><td>12-bit resolution, right-aligned</td><td>[11:00] of ADCAnCmCR[15:00]</td></tr>
<tr><td>0</td><td>1</td><td>12-bit resolution, left-aligned</td><td>[15:04] of ADCAnCmCR[15:00]</td></tr>
<tr><td>1</td><td>0</td><td>10-bit resolution, right-aligned</td><td>[09:00] of ADCAnCmCR[15:00]</td></tr>
<tr><td>1</td><td>1</td><td>10-bit resolution, left-aligned</td><td>[15:06] of ADCAnCmCR[15:00]</td></tr>
</table></td></tr>
</table>

Note If A/D conversion is performed using the internal reference voltage, the A/D conversion result is stored in ADCAnDGCR, not in the ADCAnLCR, ADCAnCmCR, and ADCAnDBiCR (see *(4) "ADCAnDGCR – Diagnostic conversion result register" on page 2392*).

(3) ADCAnDBiCR – DMA Buffer register of CGi

This register stores the result and the status of the latest A/D conversion of CGi. It allows the A/D conversion results for all channels of CGi to be read continuously.

Access This register can be read in 32-bit units.

- The upper 16 bits store the A/D conversion result status.
- The lower 16 bits store the A/D conversion result.

Address <ADCAn_base1> + $C4_H$ + i x 4_H

Initial Value 0000 0000_H + i x 0100 0000_H. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	ADCAn DBiCG[1:0]		ADCAn DBiER1	ADCAn DBiER0	ADCAn DBiUR	ADCAnDBiCN[4:0]				
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnDBiCR[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Note The functions of the individual bits are identical to those of the ADCAnLCR register, except that this register indicates the latest A/D conversion result for a specific CGi rather than for all the CGs like the ADCAnLCR register (see *Table 32-28 “ADCAnLCR register contents” on page 2387*).

Table 32-30 ADCAnDBiCR register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>25, 24</td><td>ADCAn DBiCG[1:0]</td><td>Specifies the CG to which the result in ADCAnDBiCR[15:00] belongs.
<table>
<tr><th>ADCAn DBiCG1</th><th>ADCAn DBiCG0</th><th>Channel group</th></tr>
<tr><td>0</td><td>0</td><td>CG0</td></tr>
<tr><td>0</td><td>1</td><td>CG1</td></tr>
<tr><td>1</td><td>0</td><td>CG2</td></tr>
<tr><td>1</td><td>1</td><td>None</td></tr>
</table>
The values of these bits are fixed in order to always store the same CG conversion result and status.</td></tr>
<tr><td>23</td><td>ADCAn DBiER1</td><td>Indicates the overwrite error status.
0: Not overwritten
1: Overwritten
This error flag is cleared when ADCAnSTC2.ADCAnDBiERC1 is set to 1.</td></tr>
<tr><td>22</td><td>ADCAn DBiER0</td><td>Indicates the A/D conversion result upper/lower limit comparison status.
0: The conversion result is within the setting range.
1: The conversion result is out of the setting range.
This error flag is cleared when ADCAnSTC2.ADCAnDBiERC0 is set to 1.</td></tr>
<tr><td>21</td><td>ADCAn DBiUR</td><td>Indicates the update status of the A/D conversion result.
0: The A/D conversion result is read from the ADCAnCmCR register.
1: This is the latest value, and the A/D conversion result is not read from the ADCAnDBiCR register.
This bit is cleared after the read operation.</td></tr>
<tr><td>20 to 16</td><td>ADCAn DBiCN[4:0]</td><td>Indicates the channel number corresponding to the conversion result stored in the ADCAnDBiCR[15:00].
00001 × m = CHm</td></tr>
<tr><td>15 to 0</td><td>ADCAn DBiCR[15:00]</td><td>Indicates the result of the A/D conversion.
The resolution and alignment are determined by ADCAnCTL1.ADCAnCTYP and ADCAnCTL1.ADCAnCRAC, as follows.
<table>
<tr><th>ADCAn CTL1. ADCAn CTYP</th><th>ADCAn CTL1. ADCAn CRAC</th><th>Resolution and alignment</th><th>Bit positions of A/D conversion result</th></tr>
<tr><td>0</td><td>0</td><td>12-bit resolution, right-aligned</td><td>[11:00] of ADCAnDBiCR[15:00]</td></tr>
<tr><td>0</td><td>1</td><td>12-bit resolution, left-aligned</td><td>[15:04] of ADCAnDBiCR[15:00]</td></tr>
<tr><td>1</td><td>0</td><td>10-bit resolution, right-aligned</td><td>[09:00] of ADCAnDBiCR[15:00]</td></tr>
<tr><td>1</td><td>1</td><td>10-bit resolution, left-aligned</td><td>[15:06] of ADCAnDBiCR[15:00]</td></tr>
</table></td></tr>
</table>

Note The functions of the individual bits are identical to those of the ADCAnLCR register, except that here they affect the latest result for a CGi rather than the latest A/D conversion result for all channels (see *Table 32-28 “ADCAnLCR register contents” on page 2387*).

(4) ADCAnDGCR – Diagnostic conversion result register

This register stores the A/D conversion result of the reference voltage ADDIAGOUT signal (when ADCAnCG0.ADCAnDIAG = 1).

The diagnosis conversion starts after the A/D conversion of the last channel of CG0 has been completed.

Access This register can be read in 16-bit units.

Address <ADCAn_base1> + $9C_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnDGCR[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 32-31 ADCAnDGCR register contents

Bit position	Bit name	Function
15 to 0	ADCAn DGCR[15:00]	Result of the diagnostic A/D conversion. The resolution and alignment depend on ADCAnCTL1.ADCAnCTYP and ADCAnCTL1.ADCAnCRAC (as for the “normal” A/D conversion result registers).

32.4.6 A/D conversion result upper/lower limit comparison registers

(1) ADCAnCTL2 – A/D converter result check register

This register can enable/disable the conversion result upper/lower limit comparison function for each channel.

See *32.3.12 "Result check functions" on page 2353* for details.

Access This register can be read/written in 32-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

Address <ADCAn_base1> + 18_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnRCK[23:16]							
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnRCK[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 32-32 ADCAnCTL2 register contents

Bit position	Bit name	Function
23 to 00	ADCAn RCK[23:00]	Enables/disables result upper/lower limit comparison for CHm. 0: Do not perform upper/lower limit comparison for A/D conversion of CHm 1: Perform upper/lower limit comparison for A/D conversion of CHm Note: Set the bits corresponding to channels not provided in this product to 0.

<R>
<R>

Note The settings are valid for A/D conversions of every CG.

(2) ADCAnUL – A/D converter result upper/lower limit comparison (upper limit)

This register specifies the upper limit of the A/D conversion result.

See *32.3.12 "Result check functions" on page 2353* for details.

Access This register can be read/written in 16-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

Address <ADCAn_base1> + $1C_H$

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnUL[11:00]												0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R

Table 32-33 ADCAnUL register contents

Bit position	Bit name	Function
15 to 4	ADCAnUL[11:00]	Specifies the upper limit of the A/D conversion result. In the case of the 10-bit resolution, use ADCAnUL[11:02].

(3) ADCAnLL – A/D converter result upper/lower limit comparison (lower limit)

This register specifies the lower limit of the A/D conversion result.

See *32.3.12 "Result check functions" on page 2353* for details.

Access This register can be read/written in 16-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

Address <ADCAn_base1> + 20_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnLL[11:00]												0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R

Table 32-34 ADCAnLL register contents

Bit position	Bit name	Function
15 to 4	ADCAnLL[11:00]	Specifies the lower limit of the A/D conversion result. In the case of the 10-bit resolution, use ADCAnLL[11:02].

(4) ADCAnSTR0 – A/D converter result upper/lower limit comparison error flag

This register indicates the error status of the latest A/D conversion result upper/lower limit comparison for the channel set in the ADCAnCTL2 register. By evaluating this register it is possible to deduce which A/D conversion results are outside the specified range.

See *32.3.12 “Result check functions” on page 2353* for details.

Access This register can be read in 32-bit units.

Address <ADCAn_base1> + 24_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnRCE[23:16]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnRCE[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 32-35 ADCAnSTR0 register contents

Bit position	Bit name	Function
23 to 0	ADCAn RCE[23:00]	Indicates whether the A/D conversion result is within the specified value range: 0: Conversion results within the specified range 1: At least one Conversion result out of the specified range This error flag is cleared by setting ADCAnSTC0.ADCAnRCECm to 1. Note: Set the bits corresponding to channels not provided in this product to 0.

Note ADCAnSTR0.ADCAnRCEm is mirrored by the following A/D conversion result error flag:

- Error flag in A/D converter conversion result register for channel m (ADCAnCmCR.ADCAnCmER0)

(5) ADCAnSTC0 – ADCAnSTR0 flag clear register

This register is the clear control register of ADCAnSTR0.

Access This register can be written in 32-bit units.
It is always read as 0000 $0000_{H.}$

Address <ADCAn_base1> + 30_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnRCEC[23:16]							
R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnRCEC[15:00]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 32-36 ADCAnSTC0 register contents

Bit position	Bit name	Function
23 to 0	ADCAn RCEC[23:00]	0: No function 1: Clears the corresponding ADCAnSTR0.ADCAnRCEm Note: Set the bits corresponding to channels not provided in this product to 0.

32.4.7 Diagnose functions registers

(1) ADCAnDGCTL0 – Self-diagnosis function control register 0

This register specifies the reference voltage to be applied to diagnose the A/D conversion circuit operation.

This register can be written even when ADCAnCTL0.ADCAnCE is set to 1.

Access This register can be read/written in 16-bit units.

Address <ADCAn_base> + DC_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	ADCAnPSEL[3:0]		
R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W

Table 32-37 ADCAnDGCTL0 register contents

Bit position	Bit name	Function
2 to 0	ADCAn PSEL[3:0]	Specifies the reference voltages

ADCAn PSEL2	ADCAn PSEL1	ADCAn PSEL0	ADDIAGOUT	DIAGOUT2	DIAGOUT1	DIAGOUT0
0	0	0	AV_{SS}	2/3 AV_{DD}	1/2 AV_{DD}	1/3 AV_{DD}
0	0	1	1/3 AV_{DD}	1/2 AV_{DD}	1/3 AV_{DD}	2/3 AV_{DD}
0	1	0	1/2 AV_{DD}	1/3 AV_{DD}	2/3 AV_{DD}	1/2 AV_{DD}
0	1	1	2/3 AV_{DD}	Hi-Z	Hi-Z	Hi-Z
1	0	0	AV_{DD}	2/3 AV_{DD}	1/2 AV_{DD}	1/3 AV_{DD}
1	0	1	AV_{DD}	1/2 AV_{DD}	1/3 AV_{DD}	2/3 AV_{DD}
1	1	0	AV_{DD}	1/3 AV_{DD}	2/3 AV_{DD}	1/2 AV_{DD}
1	1	1	AV_{DD}	Hi-Z	Hi-Z	Hi-Z

When Hi-Z is selected and converted, the A/D conversion result is undefined.

See *(1) “Diagnosis for A/D conversion circuit” on page 2356* for details.

(2) ADCAnDGCTL1 – Self-diagnosis function control register 1

This register specifies the channels to which a internal reference voltage is applied (instead of the analog input signal ADCAnIm).

Access This register can be written only when ADCAnCTL0.ADCAnCE is set to 0.

Address <ADCAn_base0> + $11C_H$

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnCDG[23:16]							
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCDG[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 32-38 ADCAnDGCTL1 register contents

Bit position	Bit name	Function
23 to 0	ADCAnCDGm	Specifies the input voltage: 0: Use analog input voltage ADCAnIm 1: Use the following reference voltage: DIAGOUT0 for m = 21, 18, 15, 12, 9, 6, 3, 0 DIAGOUT1 for m = 22, 19, 16, 13, 10, 7, 4, 1 DIAGOUT2 for m = 23, 20, 17, 14, 11, 8, 5, 2 Note: Set the bits corresponding to channels not provided in this product to 0.

(3) ADCAnPDCTL0 – Internal pull down resistance control register 0

This register specifies the channels to which an internal pull-down resistor for the ADCAnIm pin is to be connected. For details, see *(3) "Diagnosis for open pins" on page 2358*.

This register can be written only when ADCAnCTL0.ADCAnCE is set to 0.

Access This register can be read/written in 32-bit units.

Address <ADCAn_base0> + 120_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnPDNAm[23:16]							
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnPDNAm[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 32-39 ADCAnPDCTL0 register contents

Bit position	Bit name	Function
23 to 0	ADCAnPDNAm	Specifies whether an internal pull-down register is connected to CHm: 0: Do not connect an internal pull-down resistor 1: Connect an internal pull-down resistor Note: Set the bits corresponding to channels not provided in this product to 0.

32.4.8 Channel S/H function setting register (product dependent)

(1) ADCAnSHCTL – A/D converter channel S/H control register

This register switches on/off the power supply to the channel S/H circuit and enables/disables the channel S/H function.

This register can be written only when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

However, when diagnosing the channel S/H circuit, this register can be written even when ADCAnCTL0.ADCAnCE is set to 1.

Access This register can be read/written in 32-bit units.

Address <ADCAn_base0> + 118_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	ADCAnCPS[12:00]												
R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	ADCAnCSEL[12:00]												
R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 32-40 ADCAnSHCTL register contents

Bit position	Bit name	Function
28 to 16	ADCAn CPS[12:00]	Controls the power supply to the channel S/H circuit. 0: Power supply to channel S/H circuit off 1: Power supply to channel S/H circuit on To reduce the power consumed by the channel S/H circuit, switch off the power supply when not using the channel S/H circuit. Note: Set the bits corresponding to channels not provided in this product to 0.
12 to 0	ADCAn CSEL[12:00]	Enables/disables the channel S/H function. 0: Disable the channel S/H function. 1: Enable the channel S/H function. When diagnosing the channel S/H circuit, ADCAnSHCTL.ADCAnCSELx can be changed even when ADCAnCE is set to 1. Note: Set the bits corresponding to channels not provided in this product to 0.

32.5 Precautions on Usage

32.5.1 Channel input voltage range

Caution Use input voltages to ADCAnIm that are within the range of the ratings. If a voltage higher than AV_{DD} or lower than AV_{SS} is input, the conversion value of that channel is undefined, and the conversion values of other channels also may be affected.

32.5.2 Stopping conversion operation

If “0” is written in ADCAnCTL0.ADCAnCE during conversion, the conversion stops and no conversion result is stored into ADCAnCmCR.

32.5.3 Restrictions when using the channel S/H function

See *(2) “Restrictions when using the channel S/H function” on page 2368*.

<R>

32.5.4 Notes on application design

(1) Analog input pin (ADCAnIm)

1. Be sure to input a voltage within the rated range to the ADCAnIm pin. It is recommended to clamp this pin using a diode with a VF of 0.3 V or less to prevent a voltage higher than AV_{REFPn} or lower than AV_{REFMn} from being applied to the pin. If a voltage higher than AV_{REFPn} or a voltage lower than AV_{REFMn} is input to ADCAnIm, the conversion value of that channel becomes undefined and is not guaranteed. In addition, the conversion value of other channels may be also affected.
2. To eliminate noise, connect resistor Re between the ADCAnIm pin and the external analog signal input source, and a capacitor Ce between the ADCAnIm pin and AV_{SSn} pin.
   Value for reference: Re = T.B.D, Ce = T.B.D
3. Do not cross an analog signal line with a digital signal line or place an analog signal line in the vicinity of a digital signal line as this may degrade the A/D conversion characteristics due to noise induction.
4. It is recommended to avoid inputting and outputting a high driving current to or from a port close to the ADCAnIm pin and switching it by toggling.

(2) Wiring of power supply

To minimize the influence of switching noise of a digital circuit on the accuracy of the A/D converter, it is recommended to take the following measures.

1. Route the power line on a single side, or use as thick a pattern as possible and connect the power line in grid.
2. Insert a bypass capacitor close to the lead between a power supply pin (EV_{DD}, $OSCV_{DD}$, FV_{DD}, $PLLV_{DD}$ (product dependent), $CPUPLLV_{DD}$

(product dependent), $FRPLLV_{DD}$ (some products only, product dependent), CV_{DD} (product dependent), V_{DDM} (product dependent), V_{DDC} (product dependent), AV_{DDn}) and a ground pin (EV_{SS}, $OSCV_{SS}$, FV_{SS}, $PLLV_{SS}$ (product dependent), $CPUPLLV_{SS}$ (product dependent), $FRPLLV_{SS}$ (some products only, product dependent), CV_{SS} (product dependent), V_{SSM} (product dependent), V_{SSC} (product dependent), AV_{SSn}). A laminated ceramic capacitor of approx. 0.1 μF (T.B.D.) or a tantalum electrolytic capacitor of 4.7 μF (T.B.D.) or higher is recommended for use as this bypass capacitor.

3. It is recommended to separate the analog power line (AV_{DDn}) from the digital power lines (EV_{DD}, $OSCV_{DD}$, FV_{DD}, $PLLV_{DD}$ (product dependent), $CPUPLLV_{DD}$ (product dependent), $FRPLLV_{DD}$ (some products only, product dependent), CV_{DD} (product dependent), V_{DDM} (product dependent), V_{DDC} (product dependent)) and to supply analog power from a series regulator. To share the analog power line with the digital power lines, short-circuit the analog power supply and digital power supply at one point with an electrolytic capacitor at the power source, and separately wire the patterns on the board. In addition, insertion of a chip inductor at the inlet of the analog power is recommended. It is also recommended to ground the analog ground and digital ground at the ground source at one point with an electrolytic capacitor, and separate the wiring of the patterns on the board.

(3) Analog reference voltage input pins (AV_{REFPn}, AV_{REFMn})

Insert a bypass capacitor close to the pin lead between the AV_{REFPn} pin and AV_{REFMn} pin. A laminated ceramic capacitor of 0.1 μF (T.B.D.) or a tantalum electrolytic capacitor of 4.7 μF (T.B.D.) or higher is recommended for use as this bypass capacitor.

(4) Variation of A/D conversion result

The result of A/D conversion may vary due to fluctuations in the supply voltage and noise. Noise at an analog input pin (ADCAnIm) or reference voltage input pin (AV_{REFPn}, AV_{REFMn}) may cause an invalid conversion result.

Mitigate these variations by software processing to protect the system from adverse influences due to these variations and illegal conversion results.

The following are examples of software processing.

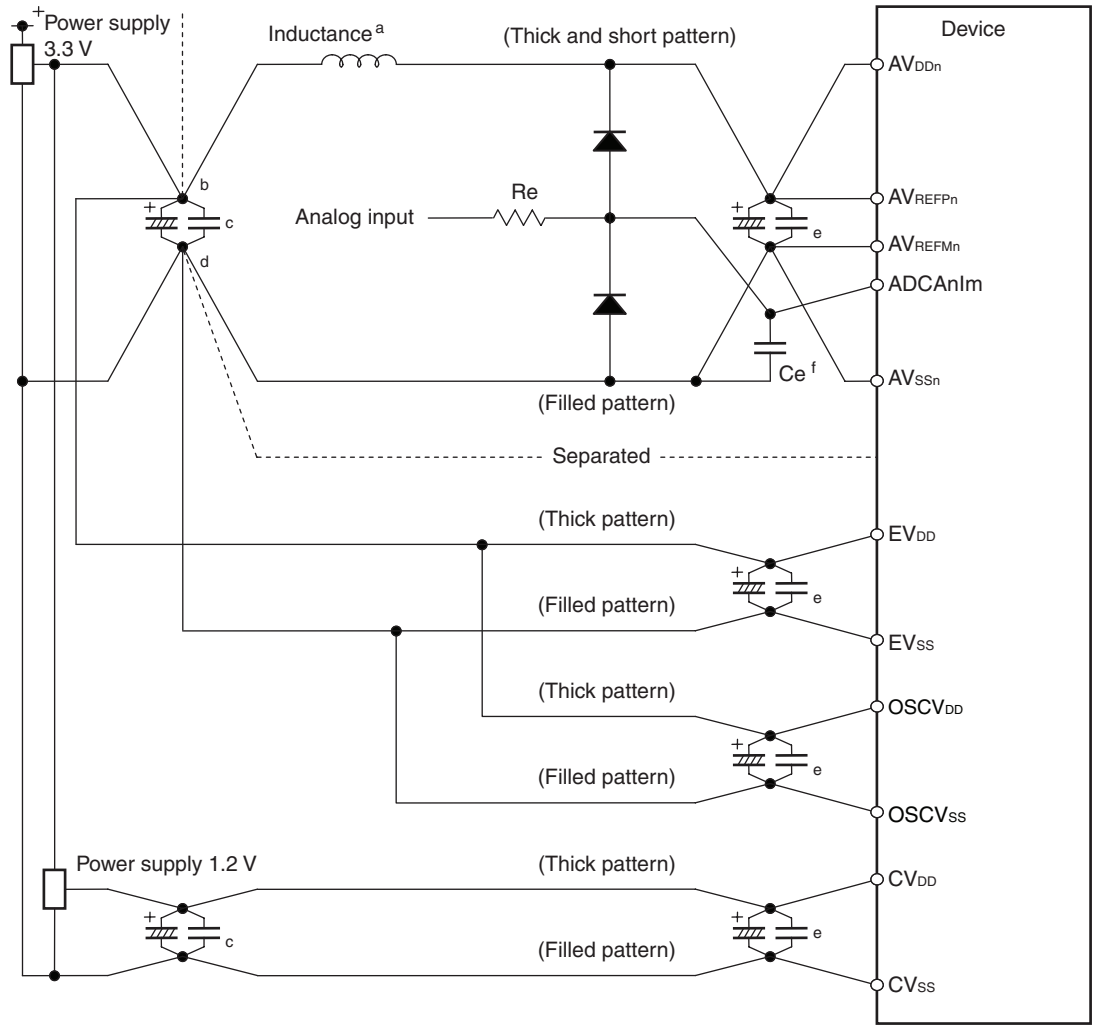
- Use the average value of the results of multiple A/D conversions as the A/D conversion result.
- Perform multiple consecutive A/D conversions and use the conversion results, with the exception of any abnormal conversion results that are obtained.
- If an A/D conversion result is obtained, from which it is judged that an abnormality has occurred in the system, do not perform abnormality processing at once but perform it upon reconfirming the occurrence of an abnormality.

(5) Hysteresis characteristics of A/D conversion

The successive approximation A/D converter holds the analog input voltage on a capacitor for the internal common sample & hold, and then executes A/D conversion. Even after A/D conversion has been completed, the analog input voltage remains on the capacitor for the internal common sample & hold. Consequently, the following phenomena may occur.

1. If the voltage rises above or falls below the voltage for the A/D conversion immediately before when A/D conversion is to be executed with a single channel, hysteresis characteristics, in which the conversion result is affected by the value immediately before, may emerge, and the conversion results may differ even at the same voltage. The hysteresis characteristics tend to increase if the signal source impedance or resistor Re of the external circuit of the analog input pin is high, or if the value of capacitor Ce is small.
2. When the analog input channel is changed, hysteresis characteristics, in which the conversion result is affected by the value of the channel immediately before, may emerge, and the conversion results may differ even at the same voltage, because A/D conversion is performed by using one A/D converter.

To obtain a more accurate conversion result, therefore, perform A/D conversion of the same channel two times in a row, and discard the first A/D conversion result.



a) Insert an inductor at the analog power inlet.
b) Short-circuit the analog power and digital power at one point in the vicinity of the 5 V power supply and separate the wiring of the patterns on the board.
c) Place a bypass capacitor (electrolytic capacitor or laminated ceramic capacitor) in the vicinity of the power supply.
d) Short-circuit the analog ground and digital ground at one point in the vicinity of the power supply and separate the wiring of the patterns on the board.
e) Place a bypass capacitor (electrolytic capacitor or laminated ceramic capacitor) in the vicinity of each power supply pin of the device.
f) Place capacitor Ce in the vicinity of the analog input pin of the device.

Note: When the analog power and digital power are shared and $AV_{REFPn} = AV_{DDn}$

<R> **Figure 32-23 Wiring example**

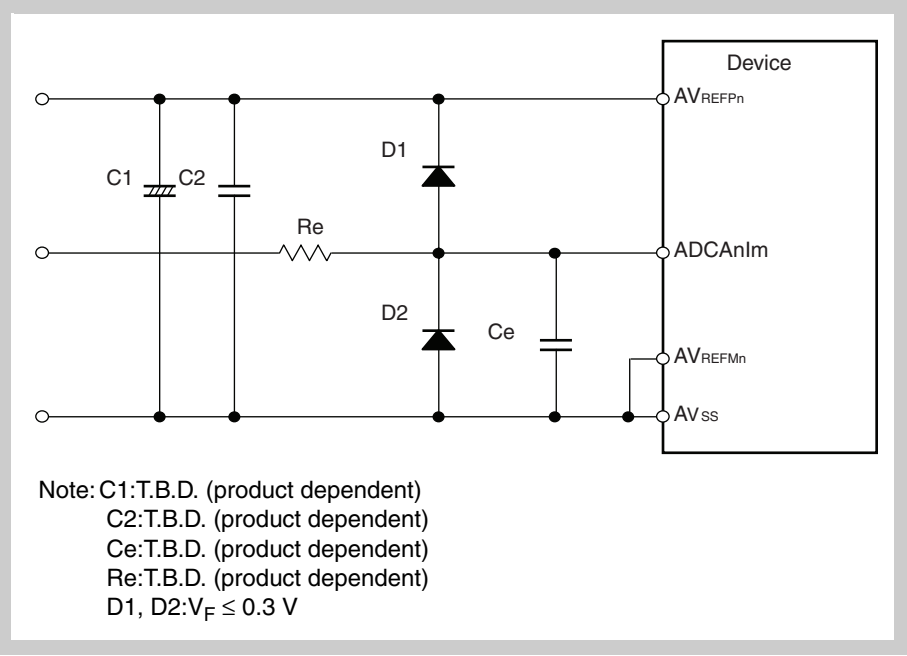


Figure 32-24 Example of countermeasures against noise on analog input circuit

Capacitor C1 is effective for eliminating low-frequency noise, and capacitors C2 and Ce are effective for eliminating high-frequency noise.

Immediately after an operation has been started with the A/D conversion stopped, the voltage applied to the AV_{DDn} and AV_{REFPn} pins is unstable, causing the accuracy of A/D conversion to drop. In this case, connect capacitors C1 and C2 to the AV_{DDn} and AV_{REFPn} pins.

32.6 How to Read A/D Converter Characteristics Table

This section describes the meanings of the terms peculiar to the A/D converter.

(1) Resolution

The minimum analog input voltage that can be identified, i.e. the ratio of the analog input voltage to 1 digital output is called 1 LSB (Least Significant Bit). The ratio of 1 LSB to the full scale is expressed as %FSR (Full Scale Range). %FSR is the ratio, in percentage, of the range in which an analog input voltage can be converted, and is expressed as follows regardless of the resolution.

1%FSR = (Maximum value of analog input voltage that can be converted − Minimum value of analog input voltage that can be converted)/100
= $(AV_{REFP} - AV_{REFM})/100$

1 LSB is as follows at a resolution of 10 bits:

1 LSB = $1/2^{10}$
= 1/1,024
= 0.098%FSR

1 LSB is as follows at a resolution of 12 bits:

1 LSB = $1/2^{12}$
= 1/4,096
= 0.024%FSR

The accuracy is determined by the total error, regardless of the resolution.

(2) Total error

This is the maximum value of the difference between the actually measured value and the theoretical value.

It is the total of the zero-scale error, full-scale error, linearity error, and a combination of these errors.

The total error shown in the characteristics table does not include the quantization error.

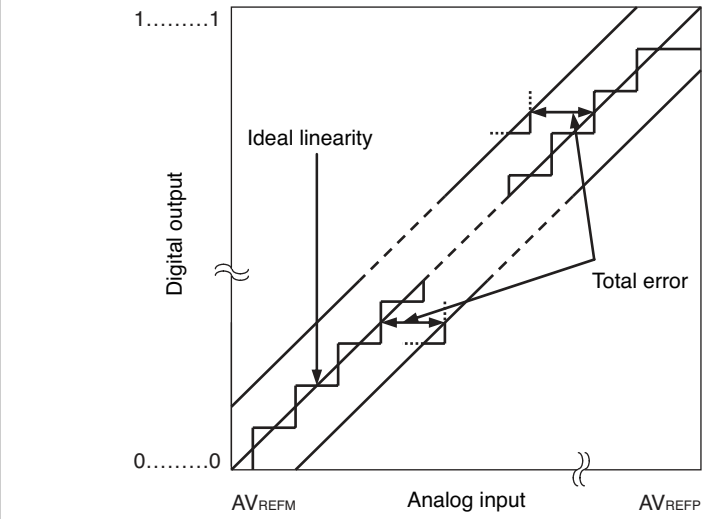


Figure 32-25 Total error

(3) Quantization error

This is the error of ±1/2 LSB that always occurs when an analog value is converted into a digital value. Because the A/D converter converts an analog input voltage in a range of ±1/2 LSB into the same digital code, the quantization error is unavoidable.

Note that this error is not included in the total error, zero-scale error, full-scale error, integral linearity error, and differential linearity error in the characteristics table.

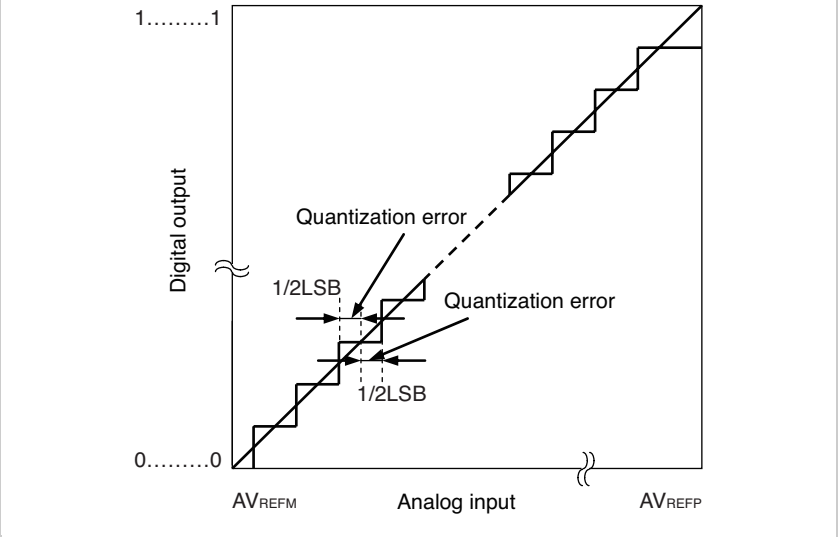
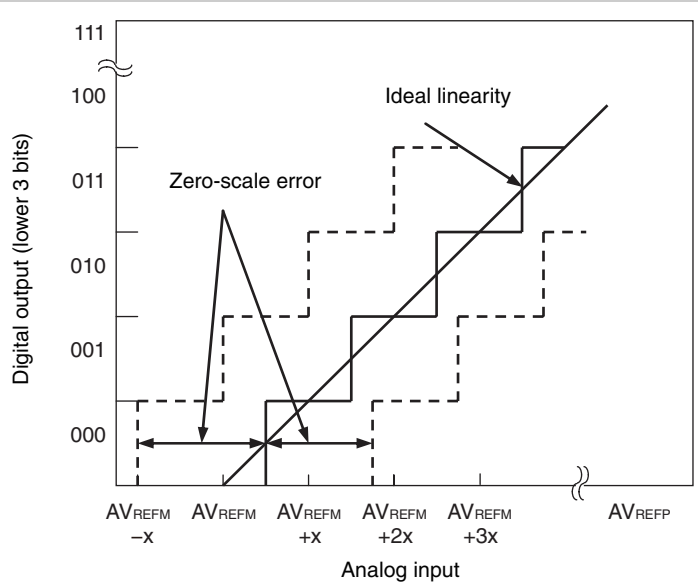


Figure 32-26 Quantization error

(4) Zero-scale error

This is the difference between the actually measured value of the analog input voltage and the theoretical value (1/2 LSB) when the digital output changes from 0…000 to 0…001.

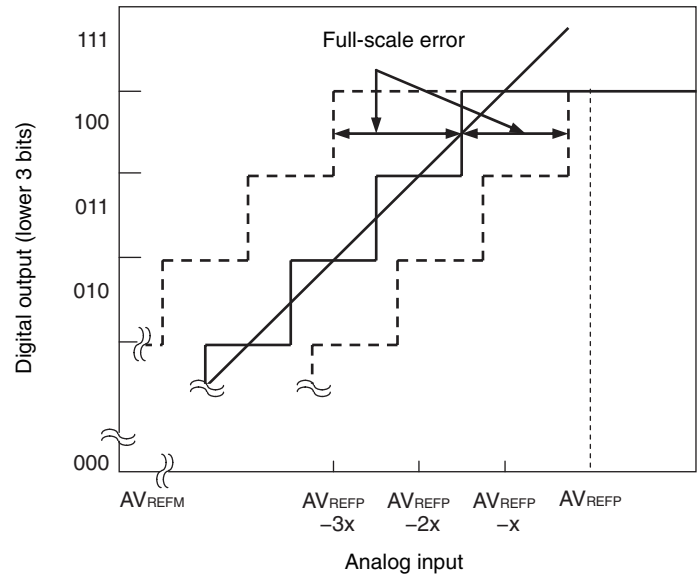


Note: x: Voltage equivalent to 1 LSB
x = (AVREP − AVREFM) × 1 LSB

Figure 32-27 Zero-scale error

(5) Full-scale error

This is the difference between the actually measured value of the analog input voltage and the theoretical value (full scale – 3/2 LSB) when the digital output changes from 1…110 to 1…111.



Note: x: Voltage equivalent to 1 LSB
x = (AVREP - AVREFM) × 1 LSB

Figure 32-28 Full-scale error

(6) Differential linearity error

Ideally, the width at which a specific code is output is 1 LSB. The differential linearity error is the difference between the actually measured value of the width at which a specific code is output and the ideal value.

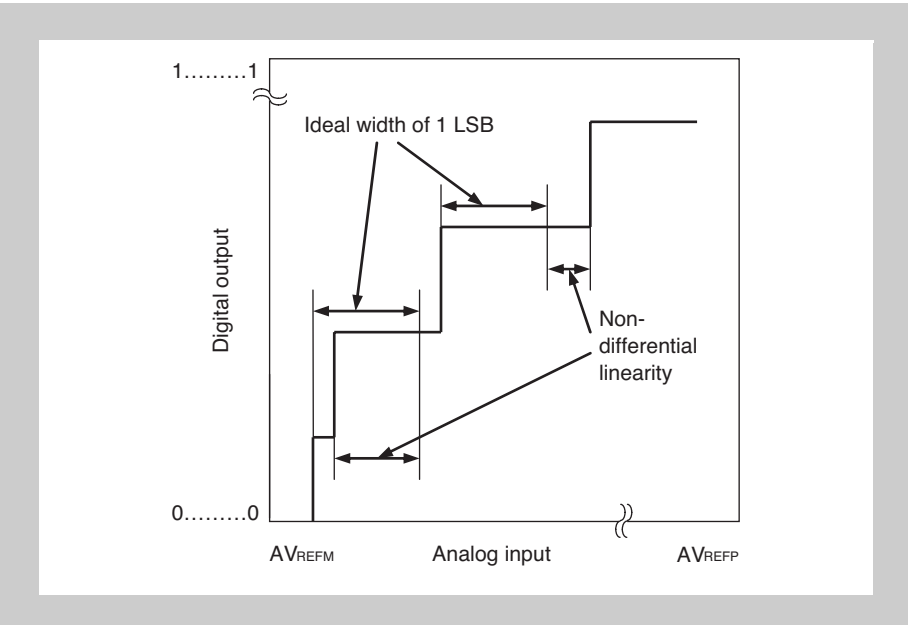


Figure 32-29 Differential linearity error

(7) Integral linearity error

This indicates the degree to which the conversion characteristic shifts from the ideal linearity, and indicates the maximum value of the difference between the actually measured value and the ideal linearity where the zero-scale error and full-scale error are 0.

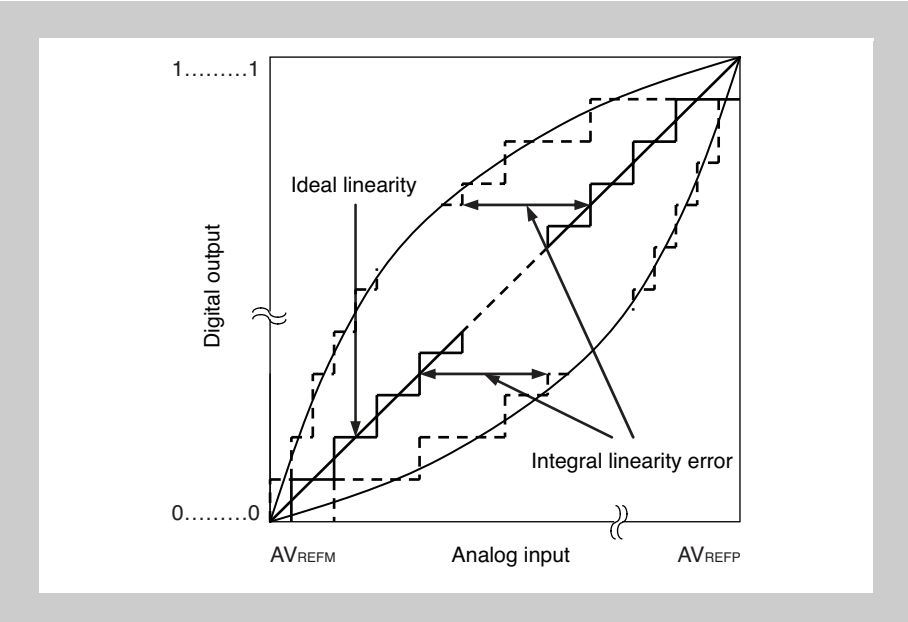


Figure 32-30 Integral linearity error

(8) Conversion time

This is the time from when an analog voltage is input until digital output is produced.

The conversion time in the characteristics table includes sampling time.

(9) Sampling time

This is the time during which the analog switch is on to input the analog voltage to the sample & hold circuit.

(10) A/D start time

This is the time from the A/D conversion trigger to the start of A/D conversion.

Chapter 33 On-Chip Debug Unit (OCD)

This microcontroller has a debug function on-chip. By using the on-chip debug emulator, programs can be debugged with the microcontroller mounted in the target system.

The debug functions incorporated in this microcontroller conform to IEEE-ISTO 5001™-2003 Class 1, a Nexus debug interface standard.

Caution The debug functions described in this chapter are supported by the microcontroller but whether they are usable depends on the debugger. For details about debugging, see the user's manual of the debugger.

33.1 V850E2/Sx4-H OCD Features

33.1.1 Modules behaviour during peripheral break

This section specifies which modules

- are always stopped (unconditional peripheral break)
- can optionally be stopped (peripheral break function)
- continue operation,

if the debugger stops the microcontroller's operation in case of an emulation break.

Peripheral break An peripheral break refers to a

- breakpoint hit
- manual break

during a debug session.

(1) Modules with unconditional peripheral break

The following list shows all modules which are always stopped upon a peripheral break:

Table 33-1 Modules with unconditional peripheral break

Module
Window watchdog timer A (WDTAn)

(2) Modules continuing operation at peripheral break

The following list shows all modules continuing operation upon an peripheral break:

Table 33-2 Modules continuing operation at peripheral break

Module
CAN controller (FCNn)
Key return function (KRn)
PCM interface (PCMn)

33.1.2 Signal masking

The V850E2/Sx4-H external signals below can be masked and disabled while the system is being controlled by the on-chip debug unit.

- $\overline{\text{RESET}}$
- NMI

33.2 Functional Overview

The on-chip debug functions are outlined below.

(1) Debug interface

This interface is used to communicate with the host by using the $\overline{\text{DCUTRST}}$, DCUTCK, DCUTMS, DCUTDI, DCUTDO, and $\overline{\text{DCURDY}}$ signals via the on-chip debug emulator.

(2) Debug monitoring function

The basic debug functions below can be used by running a monitoring program in a memory space for debugging while execution of the user-created program is paused.

- downloading the user-created program
- reading and writing the memory and registers
- running the user-created program starting at any address

(3) Hardware break function

Up to four breakpoints can be specified for data. If a breakpoint is specified for data, execution can be interrupted when data at the specified address is accessed.

In addition, break conditions can be combined by using a sequence of up to two levels.

(4) Software break function

Execution of the user-created program stored in the RAM can be interrupted at the specified address.

(5) Forced break function

Execution of the user-created program can be interrupted forcibly.

(6) Forced reset function

The microcontroller can be reset forcibly.

(7) Real-time RAM monitoring (RRM)

The memory can be read during program execution. Because this read access uses debug-dedicated DMA, it has minimal effect on program execution.

(8) Dynamic memory modification (DMM)

The memory can be written during program execution. Because this write access uses debug-dedicated DMA, it has minimal effect on program execution.

(9) Timer function

Using a 32-bit counter, the time for running the user-created program can be measured based on the clock obtained by dividing the DCUTCK signal frequency by 2.

(10) Mask function

Some dedicated external signals can be masked, so that they don't have any effect, while the microcontroller is controlled by the on-chip debug unit.

These signals are listed in *33.1.2 "Signal masking"*.

(11) Peripheral modules run/stop selection during a break

Upon a breakpoint hit the microcontroller's modules behave as follows:

- module always stops operation during break
- behaviour of the module during break is an user's option and can be specified by use of the module's emulation register.
- module always stays in operation during break

The bahaviour of the modules of this microcontroller is described in *33.1.1 "Modules behaviour during peripheral break"*.

(12) Hot plug-in function

The on-chip debug emulator can be connected and start debugging without resetting the CPU while it is running.

(13) Security function

To prevent the contents of the flash memory from being read by an unauthorized person, a 96-bit ID code can be written to the microcontroller. If the code the user inputs when starting a debugger does not match the ID code written to the microcontroller, the flash memory cannot be accessed. If the last bit (bit 95) is cleared to 0, the flash memory cannot be accessed even if the ID code matches.

For how to set the ID code, see the user's manual of the used software tools and *11.3 "On-Chip Debug Interface Protection"*.

33.3 Peripheral Break Control

The peripheral break function generates a stop request to the modules of the microcontroller, if the debugger is taking over the control of the microcontroller, for instance upon a breakpoint hit.

Those modules stop their operation during peripheral break, if

- the module supports the peripheral break function (for a list of these modules, see *33.1.1 "Modules behaviour during peripheral break"*.)
- peripheral break is enabled in the module (by setting the module's emulation register)
- peripheral break is enabled in general by setting EPC.SVSTOP = 1 (see the EPC register description below)

<R> **(1) EPC - Peripheral break control register**

This register has the functionalities that stop macros including timers, serial interfaces, and A/D converters (SVSTOP) and that prevent the disruption of access to a special sequence by register operation via the debugger.

Access This register can be read or written in 8-bit units.

Address FF43 E000$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
SV ACCESS	SV STOP	0	0	0	0	0	0
R/W	R/W	R	R	R	R	R	R

Table 33-3 EPC register contents

Bit position	Bit name	Function
7	SVACCESS	Prevents the disruption of access to a special sequence by register operation via the debugger. 0: User program accessing 1: Debugger accessing
6	SVSTOP	Stop macros including timers, serial interfaces, and A/D converters during debugging. 0: Do not stop count operation 1: Stop count operation in debug mode

33.4 Connection with On-Chip Debug Emulator

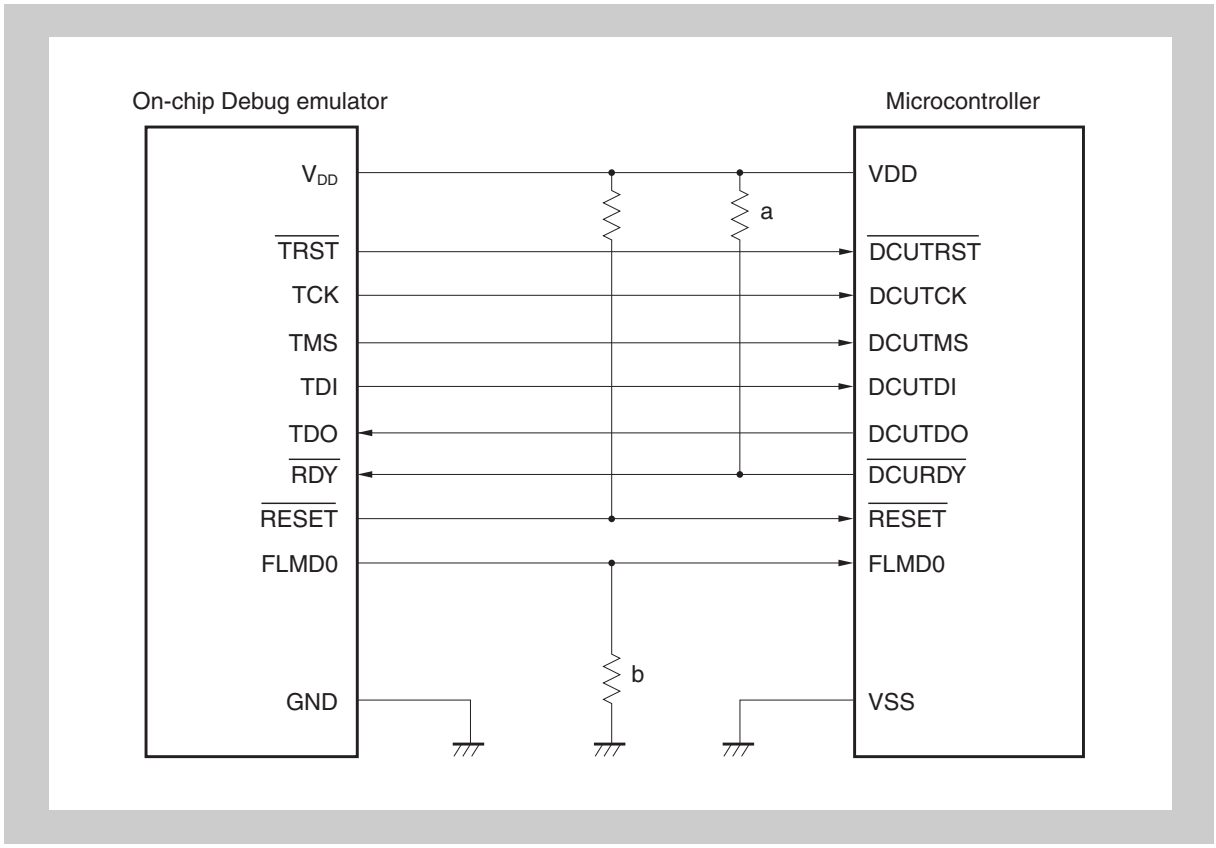


Figure 33-1 Connection with on-chip debug emulator

Notes

1. $\overline{\text{DCURDY}}$ pull-up resistor
   Since the microcontroller's $\overline{\text{DCURDY}}$ output is in high-impedance state during $\overline{\text{RESET}}$, this resistor maintains a high level at the on-chip debugger's $\overline{\text{RDY}}$ input.
2. FLMD0 pull-down resistor
   For the resistance, see the Data Sheet.

<R>

Table 33-4 Pins used to connect on-chip debug emulator

Pin name	Description
VDD	Signal used to detect power supply to the target system, or the power supply for the buffers in the on-chip debug emulator
$\overline{\text{DCUTRST}}$	Signal to asynchronously reset the debug functions of the microcontroller
DCUTCK	Clock signal used for debugging
DCUTMS	Signal to select the transfer mode for data communication
DCUTDI	Data signal input to the microcontroller
DCUTDO	Data signal output from the microcontroller
$\overline{\text{DCURDY}}$	Synchronization signal for data communication
$\overline{\text{RESET}}$	Microcontroller reset signal. Connect this pin to hold the microcontroller in the reset state from when the system is turned on until the debugger starts.
FLMD0	Mode signal used to rewrite the flash memory in the microcontroller

33.5 Cautions on Using On-Chip Debugging

(1) Handling of device that was used for debugging

Do not mount a device that was used for debugging on a mass-produced product, because the flash memory was rewritten during debugging and thus the number of flash memory rewrites cannot be guaranteed.

Chapter 34 Power Supply

The V850E2/Sx4-H devices provide separate power supply pins for the digital and analog circuits of the functional modules on the different power domains and for several groups of port I/O buffers.

At first an overview is given describing which power supply pins are used for supplying the different circuits.

It is then described how the various power supply sources have to be handled for initial power-up and final power-down of the devices and how to proceed in DEEPSTOP mode.

34.1 Naming of Power Supply Pins

In this section, some general information is given about the naming of the power supply pins.

In general, the name of a power supply pin is composed of up to four fields with the following meanings:

Supply type	Prefix	Kind of supply	Suffix
Symbol, representing the purpose of the supply	Consecutive number for separate supplies[a]	VDD or VSS	Consecutive number for different pins of the same supply[a]

a) The prefix and suffix numbers may be omitted if only a single supply pin is used.

The supply types are defined in the following table.

Table 34-1 Supply type symbols

Symbol	Description
A	Analog circuit supply (e.g. the analog sections of the A/D converter)
B	Standard I/O buffer supply
C	Core supply for digital circuits
E	Standard I/O buffer supply
F	Flash memory supply
MLB	MediaLB I/O buffer supply
OSC	Oscillator supply
REG	Internal voltage regulator input voltage

Examples:

– E0VDD: Supply for standard I/O buffer group 0 via a single pin

– REG0VSS: Input voltage for on-chip voltage regulator REG0

34.2 Power Supply Scheme

The following sections show the power supply scheme, including the power supply pins and the modules they supply.

For details about the range of the power supply pin voltage and all related conditions, see the *Data Sheet.* The *Data Sheet* also provides details about the electrical characteristics of the port pins.

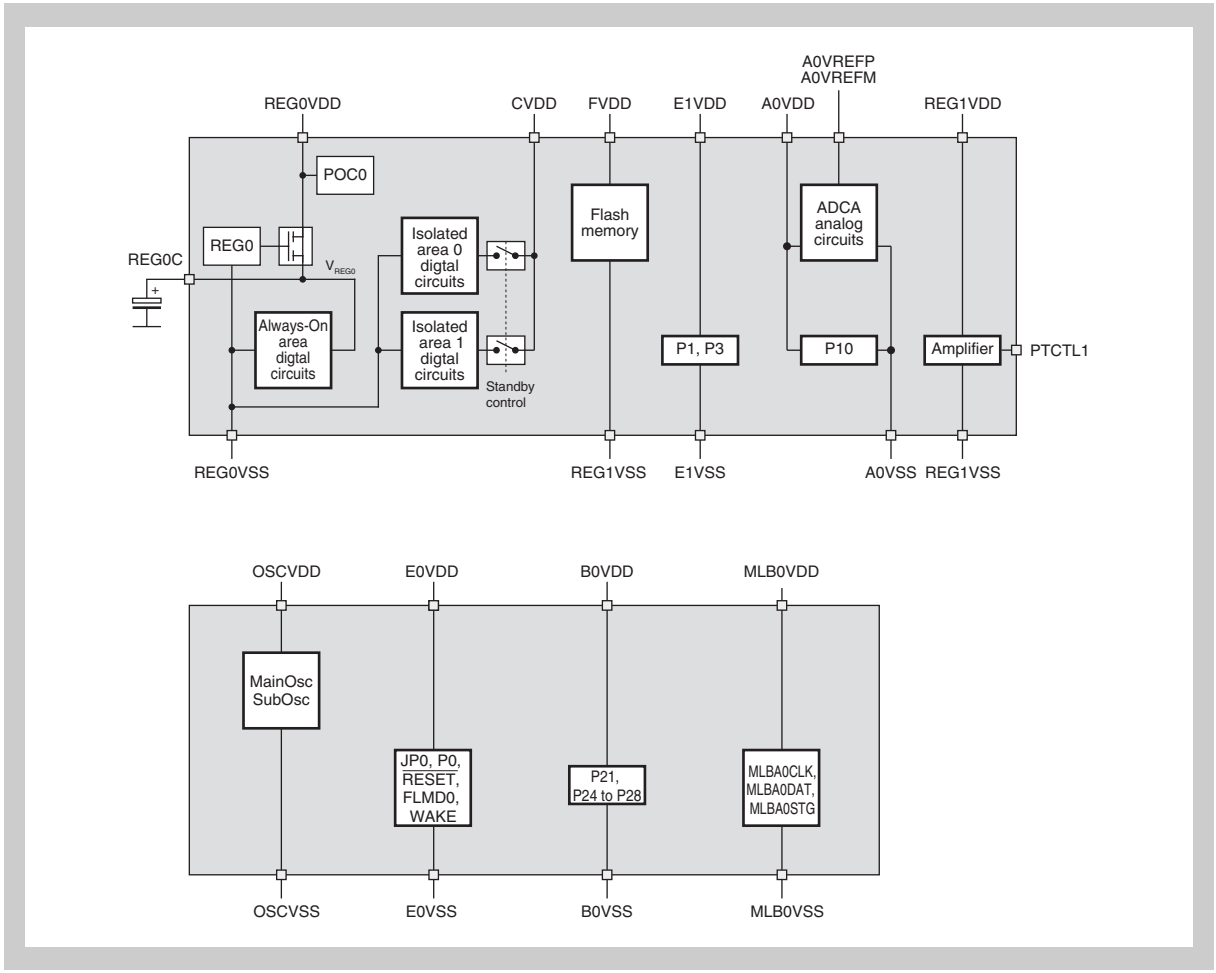
34.2.1 Power supply scheme

The following table and diagram show the power supply scheme.

Table 34-2 Power supply pins

	Pins	Module	Port buffers
	REG0VDD/REG0VSS/ REG0C	Internal voltage regulator REG0 V_{REG0} supplies the digital circuits of Always-On area modules.	–
	REG1VDD/REG1VSS	On-chip amplifier for external power transistor	PTCTL1
	CVDD/CVSS	CVDD supplies the digital circuits of Isolated area 0 and Isolated area 1 modules	–
<R>	E0VDD/E0VSS	–	JP0, P0, $\overline{\text{RESET}}$, FLMD0, WAKE
	E1VDD/E1VSS	–	P1, P3
	OSCVDD/OSCVSS	MainOsc, SubOsc	X1, X2, XT1, XT2
	B0VDD/B0VSS	–	P21, P24 to P28
<R>	A0VDD/A0VSS/ A0VREFP/A0VREFM	Analog circuits of A/D converter ADCA	P10
	MLB0VDD/MLB0VSS	–	MLBA0CLK, MLBA0DAT, MLBA0SIG
<R>	FVDD	Flash memory	–

<R>



<R> **Figure 34-1 Power supply scheme**

34.3 Power-up and Power-down Procedures

This section describes the power-up and power-down procedures.

Caution All figures are for explanatory purposes only, and do not apply to specific hardware implementations.

For details about the voltage levels, time, and frequency values, see the *Data Sheet.*

The following diagram shows an overview of the power sequencer.

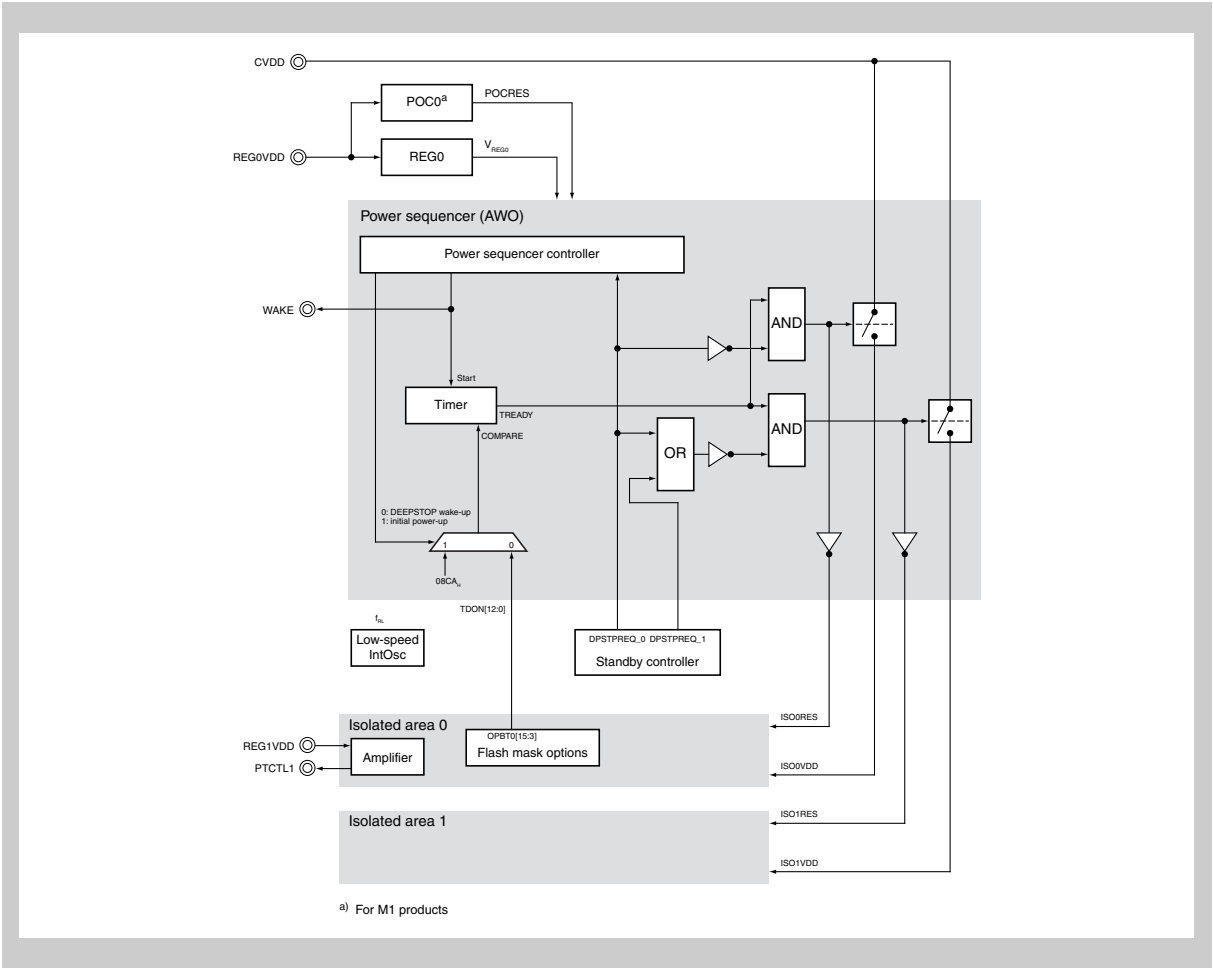


Figure 34-2 Overview of power sequencer

(1) Always-On area supply

The supply voltage V_{REG0} of the Always-On area is generated by the on-chip voltage regulator REG0 from the input voltage REG0VDD.

For M1 products, REG0VDD is monitored by the power-on-clear circuit POC. This allows the Always-On area to be kept in the reset state (the POCRES signal stays asserted) as long as the REG0VDD voltage does not falls below the POC level V_{POC}.

(2) Isolated area supply

The Isolated area supply voltages ISO0VDD and ISO1VDD are supplied by the external CVDD pin.

The Isolated areas must be powered up and powered down in the following two situations:

- Initial power-up and final power-down
  When the entire microcontroller is powered up and down, including the Always-On area.
- When entering and waking up from DEEPSTOP mode.
  In the DEEPSTOP mode, only the power supplies of the Isolated areas (ISO0VDD and ISO1VDD) are turned off; the power supply of the Always-On area remains active.
  Entering and waking up from DEEPSTOP mode is controlled by a standby controller signal.
  - DPSTPREQ_0: Asserted in DEEPSTOP mode.
  - DPSTPREQ_1: Asserted in Isolated area 1 DEEPSTOP mode.

34.3.1 Power sequencer

The power sequencer controls the power-up and power-down of Isolated areas 0 and 1.

The power supply sequencer is assigned to the Always-On area and starts operating when the power supply V_{REG0} of the Always-On area is stabilized and the power-on clear reset signal POCRES is deasserted.

To control the Isolated areas, the power sequencer:

- Turns on the Isolated area power supplies ISO0VDD/ISO1VDD if the Isolated area source supply CVDD is stable, and turns off ISO0VDD/ ISO1VDD if CVDD is not stable.
- Controls the Isolated area reset ISO0RES/ISO1RES to avoid operation of the Isolated areas when their power supply is not stable

The power sequencer assumes the external Isolated area power supply CVDD is stable, if:

- The power sequencer timer time has elapsed.
  The power sequencer timer has secured the minimum power-up time T_{PUMIN}.

(1) Power sequencer timer

The power sequencer is equipped with a timer that counts the low-speed internal oscillator period based on f_{RL} up to a certain compare value and generates the TREADY signal when this value is reached.

The timer count time is

$$T_{PU} = COMPARE \times 1/f_{RL}$$

Thus the value specified as the compare value is the minimum power-up time T_{PUMIN} from when POCRES is deasserted to when the Isolated area is powered up.

COMPARE The compare value depends on the kind of power-up:

- Initial power-up
  The compare value is fixed to $08CA_H$ = 2,250. At initial power-up, the low-speed internal oscillator is not trimmed, and with COMPARE = 2,250, the initial run time T_{PUINIT} of the power sequencer timer during initial power-up is about 11.8 ms.
- Wake-up from DEEPSTOP mode
  The compare value is specified by using TDON[12:0].
  TDON[12:0] is a flash mask option that can be specified by the user.
  The wake-up time from DEEPSTOP mode can therefore be adjusted to the application environment, and thus minimized.

(2) WAKE signal

In DEEPSTOP mode, that is when both Isolated area supplies ISO0VDD and ISO1VDD are turned off, CVDD can also be turned off.

The WAKE signal that is output to the external isolated area voltage supply CVDD indicates the following:

- If WAKE is high level, supply CVDD and keep it stabilized.
- If WAKE is low level, CVDD can be turned off.

(3) PTCTL1 signal

CVDD can also be supplied from an external power transistor. By supplying power to REG1VDD, the PTCTL1 signal, which is an output signal of the on-chip amplifier, controls the power transistor.

Caution For M1 products (with a POC function), leave the PTCTL1 pin open and do not control the PTCTL1 signal by using an external power transistor.

34.3.2 Initial power-up and final power-down

The following diagram outlines the scheme of the power supply for initial power-up and final power-down.

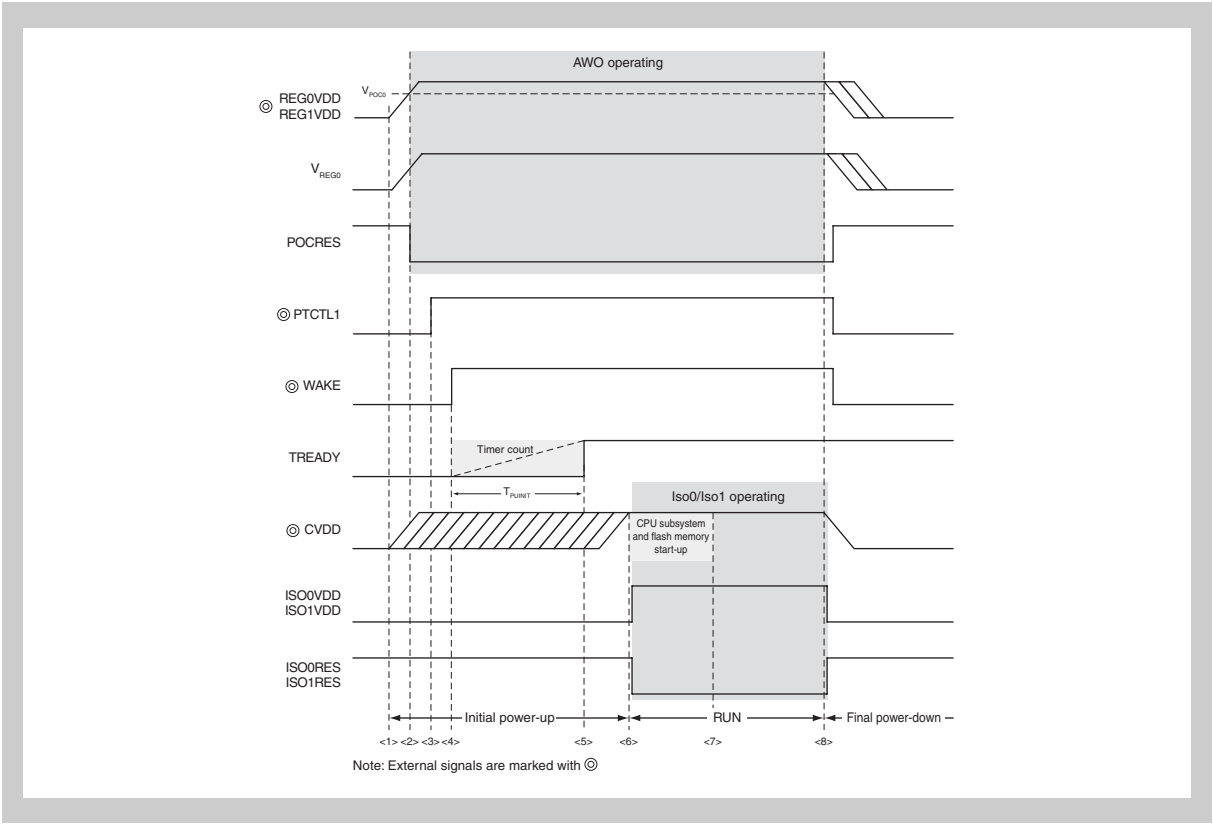


Figure 34-3 Initial power-up and final power-down

<1> The voltage regulator input REG0VDD rises, while the POCRES keeps the microcontroller in the reset state. The Always-On area voltage supply V_{REG0} starts rising.

<2> If REG0VDD exceeds the POC voltage level V_{POC}, POCRES is deasserted and the Always-On area starts operating.

<3> When REG1VDD is supplied, output of the PTCTL1 signal for controlling the external power transistor starts.

<4> The power sequencer starts its timer and asserts the WAKE signal to start the Isolated area power-up process.
CVDD rises.

<5> The power sequencer initial power-up time T_{PUINIT} has elapsed.

<6> If CVDD is stable, the power supply sequencer powers up ISO0VDD/ ISO1VDD, and then deasserts ISO0RES/ISO1RES.
The Isolated areas are operating.

<7> After startup of the CPU subsystem and the flash memory, the CPU fetches its first instruction.

<8> Final power-down stops all power supplies.

Cautions

1. At initial power-up, the external Isolated area supply CVDD must not be applied before the REG0VDD voltage for the Always-On area.
2. At final power-down, the Always-On area supply REG0VDD must not be switched off before the Isolated area supply CVDD.

For details about the timing relationship between REG0VDD and CVDD, see the Data Sheet.

34.3.3 DEEPSTOP entry and wake-up

In DEEPSTOP mode, the power supply of the Isolated areas is switched off.

- DEEPSTOP mode:
  Power supply of Isolated area 0 and Isolated area 1 is switched off
- RUN mode (Iso1 DEEPSTOP) and STOP mode (Iso1 DEEPSTOP):
  Power supply of Isolated area 1 is switched off

In either case, the Always-On area remains active.

The standby controller generates indication signals. These signals depend on the standby mode.

- DPSTREQ_0 indicates a DEEPSTOP request for Isolated area 0 and Isolated area 1.
- DPSTREQ_1 indicates a DEEPSTOP request for Isolated area 1.

(1) Voltage regulator control in DEEPSTOP mode

A low-level WAKE signal indicates that the internal Isolated area voltage supplies ISO0VDD/ISO1VDD are switched off in DEEPSTOP mode, and thus the external voltage supply CVDD can be switched off as well.

Switching off the voltage regulator during DEEPSTOP reduces power consumption, but lengthens the wake-up time, because the entire power-up sequence has to be executed.

To minimize the wake-up time, the external CVDD supply can be kept active in DEEPSTOP mode by suppressing the assertion of the WAKE signal.

Consequently, the power sequencer can switch on the Isolated area power supplies ISO0VDD/ISO1VDD immediately after wake-up.
However keeping the regulator active during DEEPSTOP increases the power consumption.

The PSC1.PSC1REGSTP bit determines the behavior of the WAKE output signal in DEEPSTOP mode:

- PSC1.PSC1REGSTP bit = 0:
  - WAKE remains high level in DEEPSTOP mode.
  - CVDD must not be switched off.
  - The power-up sequence is not executed at wake-up, and ISO0VDD/ ISO1VDD are switched on immediately.
- PSC1.PSC1REGST bit = 1:
  - WAKE changes to low level in DEEPSTOP mode.
  - CVDD can be turned off.
  - The power-up sequence is executed at wake-up (by the power sequencer timer).

For details, see *10.3.2 (2) "PSC1 – Power save control register 1"*.

If the external Isolated area voltage supply CVDD is switched off during DEEPSTOP, the power sequencer timer run time is:

$$T_{PUMIN} = TDON[12:3] \times 1/f_{RL}$$

For details about the minimum f_{RL} and maximum f_{RL} values, see the *Data Sheet*.

The following diagram shows the principle DEEPSTOP entry and wake-up flow, when the Isolated area power supply source CVDD is switched off during DEEPSTOP (PSC1.PSC1REGSTP = 1).

The following description assumes complete DEEPSTOP mode, that is, both Isolated area power supplies are switched off. However, it is also valid for RUN mode (Iso1 DEEPSTOP) in which DPSTPREQ_0, ISO0VDD, and ISO0RES are not involved.

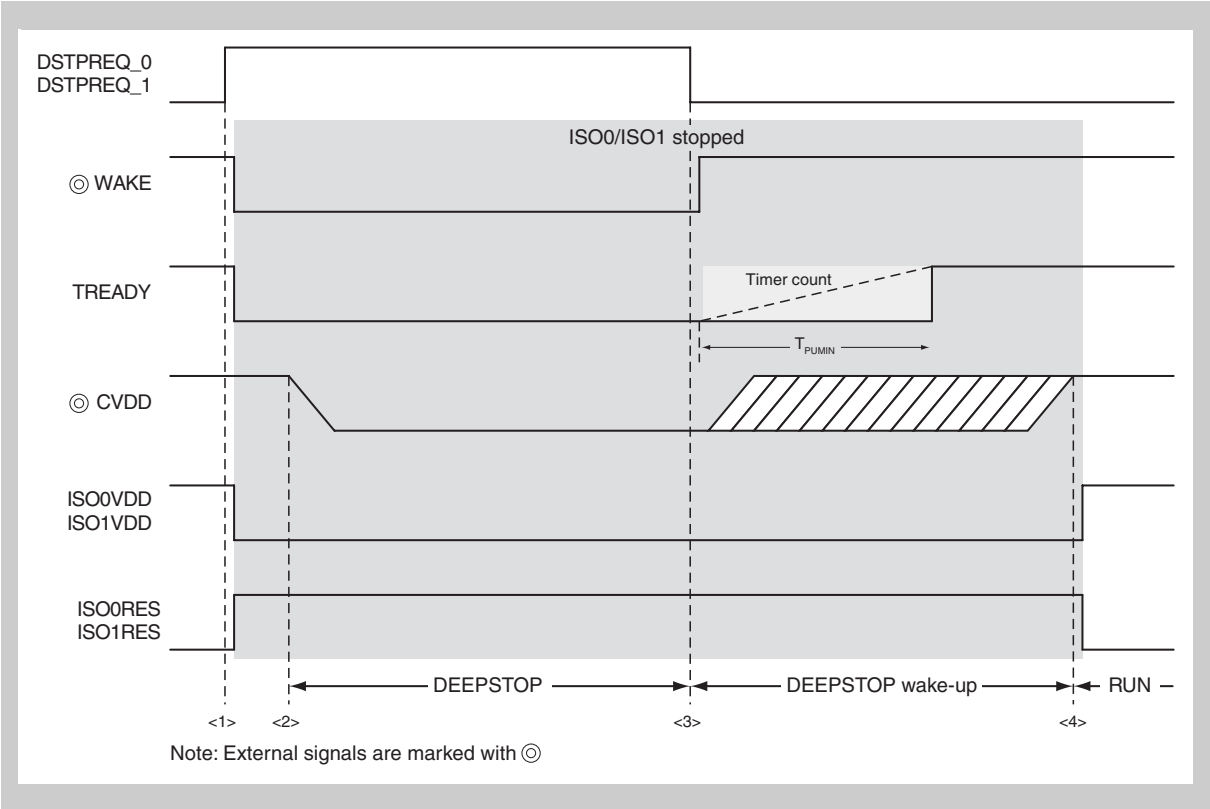


Figure 34-4 DEEPSTOP entry and wake-up (PSC1.PSC1REGSTP bit = 1)

(1) When a DEEPSTOP request is issued (DPSTPREQ_0 = DPSTPREQ_1 = 1):

- The WAKE signal is deasserted.
- The Isolated area voltage supplies ISO0VDD/ISO1VDD are switched off and set to their reset state (ISO0RES = ISO1RES = 1).

(2) The external voltage CVDD can be switched off.

(3) A wake-up event deasserts DPSTPREQ_0/DPSTPREQ_1, causing the power sequencer to start the Isolated area power-up process by asserting the WAKE signal and starting the timer.

The count time T_{PU} is determined by the TDON[12:0] value, which is specified by using the flash mask option OPBT0.OPBT0[15:3].
CVDD must be switched on and stabilized.

(4) The power sequencer wake-up time T_{PU} has elapsed, CVDD is stable, the Isolated area supplies ISO1VDD/ISO2VDD are switched on and the ISO1RES/ISO2RES reset signals are deasserted.

34.3.4 Other power supplies

This section describes how to handle the power supplies that are not used to power the Always-On area and the Isolated areas.

The following table summarizes all power supplies and their power-up and power-down conditions and options.

Table 34-3 Power supplies

Pin	Supply	Condition
REG0VDD	• Away-On area digital circuits	• All must be supplied simultaneously. • Must not be stopped in DEEPSTOP mode.
REG1VDD	• Amplifier for controlling external power transistor	
E0VDD	• Port group JP0 and P0 buffers	
OSCVDD	• Main oscillator	• Must not be supplied before REG0VDD. • Must be supplied and stable if MainOsc is active. • May be stopped in DEEPSTOP mode.
AVDD	• ADCA0 analog circuit • Port group P10	• Must not be supplied before REG0VDD. • Must be supplied and stable if ADCA0 is used. • May be stopped in DEEPSTOP mode.
CVDD	• Isolated area digital circuits	• Must not be supplied before REG0VDD. • May be stopped in DEEPSTOP mode.
FVDD	• Flash memory	• Must not be supplied before REG0VDD. • Must not be stopped in DEEPSTOP mode.
<R> B0VDD	• Port groups 21 and 24 to 28	
MLB0VDD	• MLBA0SIG, MLBA0DAT, MLBA0CLK	
E1VDD	• Port group P1 and P3 buffers	

Note For details about the handling of power supply pins, see the *Data Sheet.*

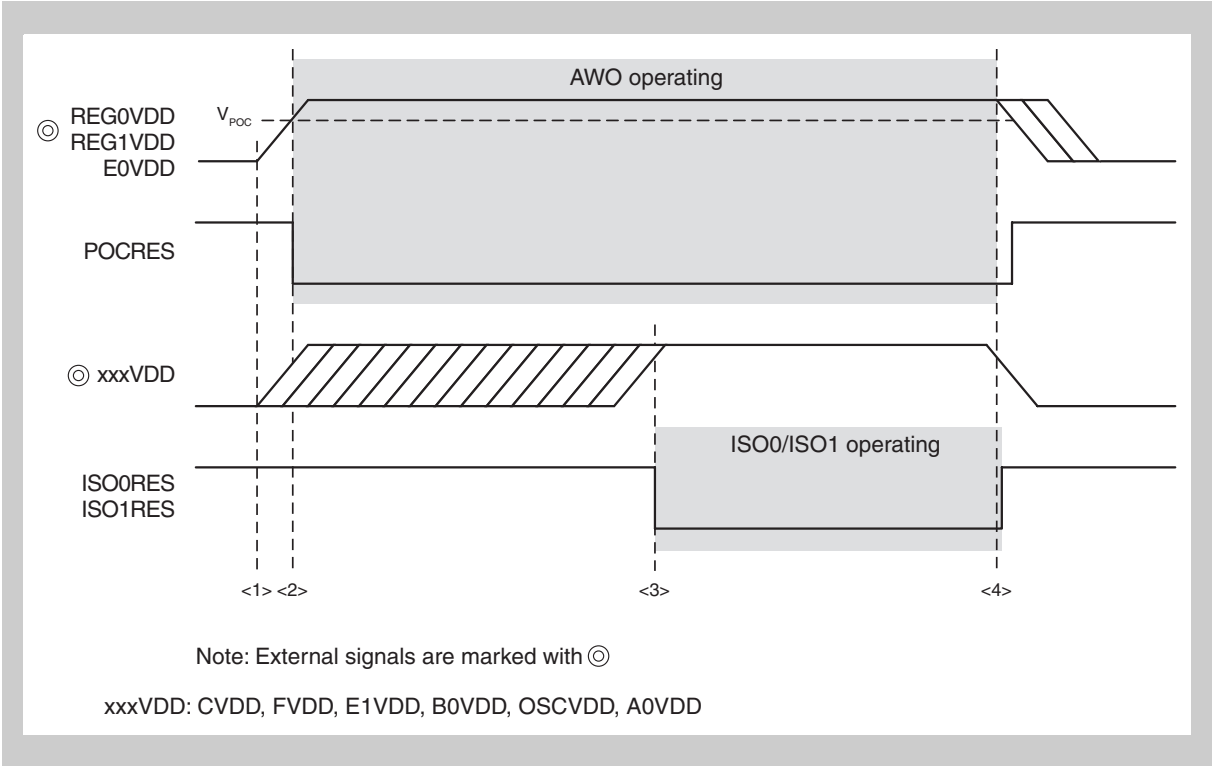


Figure 34-5 Other power supplies

<1> The voltage regulator inputs REG0VDD and REG1VDD, and the port power supply E0VDD rise, while the POCRES keeps the microcontroller in the reset state.

<2> If REG0VDD exceeds the POC voltage level V_{POC}, POCRES is deasserted and the Always-On area is activated.
The power sequencer starts the Isolated area power-up process, as described in *34.3.1 "Power sequencer"*.

<3> If xxxVDD is stable, the Isolated areas are powered up and the ISO0RES/ISO1RES reset signals are deasserted.

<4> For final power down of the entire microcontroller, xxxVDD and xxxVDD are switched off, causing the isolated areas to be reset (ISO0RES/ISO1RES = 1) and switched off.

REG0VDD and REG1VDD can be switched at the same time as CVDD, but not before.

If REG0VDD drops below V_{POC0}, the entire microcontroller is reset (POCRES = 1) and the Always-On area is switched off.

Revision History

Rev.	Date	Description	
		Page	Summary
0.01	Feb. 10, 2011	–	First edition issued
0.02	Sep. 30, 2011	Throughout	Change of the following items: • Number of I/O ports • Number of A/D cannels and URTE channels • Part numbers • Pin names
		27, 28	Modification of description in *Table 1-1 V850E2/SG4-H products*
		29	Modification of *Figure 1-1 Block diagram of V850E2/SG4-H*
		30	Modification of description in *Table 1-2 V850E2/SJ4-H products*
		32	Modification of *Figure 1-2 Block diagram of V850E2/SJ4-H*
		36	Modification of description in *Table 1-5 V850E2/Sx4-H ordering information*
		38	Modification of figure in *1.5.3 V850E2/SK4-H*
		39, 40, 42	Modification of description in *Table 1-6 Pin assignment*
		67	Deletion of description from *Table 2-22 PDSCn register contents*
		70	Modification of description in *2.3.5 (2) PPROTSn - Port protection status register*
		77	Modification of description in and addition of description to *2.4.2 (1) P0_0: $\overline{\text{RESETOUT}}$*
		78	Addition of description to *Table 2-29 Permanent input connection pins*
		82, 83, 84	Modification of description in and deletion of description from *Table 2-31 V850E2/SG4-H general-purpose I/O operations*
		85, 86	Modification of description in *Table 2-32 V850E2/SG4-H port control registers (groups 0, 1, 10, 21)*
		87, 88	Modification of description in *Table 2-33 V850E2/SG4-H port control registers (groups 25, 27, JP0)*
		89, 90, 92	Modification of description in *Table 2-34 V850E2/SJ4-H general-purpose I/O operations*
		93, 94	Modification of description in *Table 2-35 V850E2/SJ4-H port control registers (groups 0, 1, 3, 10)*
		95	Modification of description in *Table 2-36 V850E2/SJ4-H port control registers (groups 21, 25 to 27)*
		99, 100, 101, 102	Modification of description in *Table 2-38 V850E2/SK4-H general-purpose I/O operations*
		105	Modification of description in *Table 2-40 V850E2/SK4-H port control registers (groups 21, 24 to 26)*
		110, 118, 136	Modification of description in *Table 2-42 List of pin functions in alphabetical order*
		137	Addition of description to *2.4.8 Recommended connection of unused pins*
		142, 143	Modification of description in and addition of description to *Table 2-46 Input signals and control registers for ports that incorporate digital filter type D*
		149	Modification of description in *2.6.2 Digital filters*
		166	Modification of figure in *Figure 3-4 V850E2/Sx4-H CPU subsystem*

Rev.	Date	Description	
		Page	Summary
0.02	Sep. 30, 2011	182	Addition of description to *3.7.3 (3) Internal local RAM area*
		182	Addition of description to *3.7.3 (4) Internal backup RAM area*
		187	Deletion of description from *3.9 HBUS Bridge in CPU Subsystem*
		187	Deletion of description from *Figure 3-14 HBUS bridge in CPU subsystem*
		188, 189	Deletion of description from *3.9.1 (1) HBUS master interface (for transferring data from the CPU subsystem to HBUS)*
		191	Modification of description in *3.9.3 (1) ETACFG - Configuration register*
		191	Deletion of description from *Table 3-32 ETACFG register contents*
		192	Modification in and addition to *3.9.3 (2) ETACMD - Command register*
		192	Deletion of description from *Table 3-33 ETACMD register contents*
		194	Modification of description in *Table 3-35 ETARCFGn register contents*
		206	Modification of description in *3.10.6 (1) PROTCMDn - Protection command register*
		206	Modification of description in *Table 3-43 PROTCMDn register contents*
		210	Addition of *3.11 System Error Report Configuration Registers*
		231	Addition of description to *4.2.11 SDCR - SDRAM configuration register*
		241, 242	Addition of *Table 4-19 Address bus pins connected to SDRAM*
		273	Modification of *Figure 4-27 SRAM connection example*
		274	Modification of *Figure 4-28 512 Mb SDRAM connection example (256-bit SDRAM (4M words × 16 bits × 4 banks) × 2)*
		278	Modification of *Figure 4-35 Multiplexed bus mode connection example*
		332	Modification of description in *Table 6-2 DMA start sources (0 to 63)*
		363	Deletion of description from *Table 6-18 DDCn register contents*
		375	Modification of *Figure 6-2 Example of single transfer (8/16/32 bits)*
		375	Modification of *Figure 6-3 Example of single transfer (128 bits, DMA channel priority order: CHO (high) > CH1 (low))*
		376	Modification of *Figure 6-4 Example of single-step transfer (8 bits, DMA channel priority order: CHO (high) > CH1 (low))*
		376	Modification of *Figure 6-5 Example of single-step transfer (128 bits, DMA channel priority order: CHO (high) > CH1 (low))*
		411	Modification of description in *Table 7-11 Relationship between boot blocks and boot swap cluster*
		429	Addition of description to *9.2 (1) Clock generators*
		436	Deletion of description from *9.3.3 High-speed internal oscillator (high-speed IntOsc) clock generator*
		444 to 446	Addition of description to *9.4.1 Clock domains of the Always-On area*
		447 to 451	Addition of description to *9.4.2 Clock domains of Isolated area*
		452 to 469	Addition of description to *9.4.3 Clock domains of Isolated area*
		473	Modification of description in *Table 9-7 MOSCC register contents*
		479	Modification of description in *9.5.2 (8) ROSCE - High-speed IntOsc enable register*
		479	Deletion of description from *Table 9-12 ROSCE register contents*
		491	Deletion of description from *9.6.2 CLMA enable*

Rev.	Date	Description	
		Page	Summary
0.02	Sep. 30, 2011	492	Deletion of description from *9.6.3 Functional overview*
		492	Modification of *Figure 9-9 Block diagram of clock monitor A*
		493	Modification of description in *9.6.4 Description*
		496	Deletion of description from *Table 9-27 Clock monitor registers*
		496	Deletion of description from *9.6.5 (1) CLMAnCTL0 - CLMAn control register 0*
		497	Deletion of description from and addition of description to *Table 9-28 CLMAnCTL0 register contents*
		497	Deletion of description from *9.6.5 (2) CLMAnCMPL - CLMAn comparison register L*
		498	Modification of description in and addition of description to *Table 9-29 CLMAnCMPL register contents*
		498	Deletion of description from *9.6.5 (3) CLMAnCMPH - CLMAn comparison register H*
		498	Deletion of description from *9.6.5 Clock monitor registers*
		499	Modification of description in and addition of description to *Table 9-30 CLMAnCMPH register contents*
		499	Addition of description to *Table 9-30 CLMAnEMU0 register contents*
		500	Addition of description to *Table 10-1 STBC reset signal*
		505	Deletion of description from *Table 10-5 Power-save modes overview*
		505	Modification of description in and deletion of description from *Table 10-6 Clock sources in power-save modes*
		507	Deletion of description from *10.2.1 (1) Wake-up sources*
		509	Deletion of description from *10.2.2 I/O buffer control*
		510	Modification of description in *Table 10-8 Buffer operation in DEEPSTOP mode and after wake-up*
		511	Modification of figure in *Figure 10-1 Power save mode transitions*
		512	Modification of description in *10.2.4 Examples of entering and exiting power save mode*
		513	Deletion of description from *Figure 10-2 Recommended flow for entering and exiting STOP mode*
		515	Modification of figure in *Figure 10-3 Recommended flow for entering and exiting RUN mode (Iso1 STOP)*
		516	Modification of description in *10.2.4 (3) DEEPSTOP mode*
		517	Modification of figure in *Figure 10-4 Recommended flow for entering and exiting DEEPSTOP mode*
		536	Modification of description in *Table 11-1 On-chip debug protection levels*
		538	Modification of description in *11.3.4 (2) IDMODI - On-chip debug control register*
		548	Modification of description in *12.2.4 RAM retention voltage indicator (RAMHF)*
		557	Deletion of description from *Table 12-5 LVICNT register contents*
		558	Modification of description in *12.3.5 (1) VLVF - RAM retention voltage detection flag register*
		558	Modification of description in *Table 12-6 VLVF register contents*

Rev.	Date	Description	
		Page	Summary
0.02	Sep. 30, 2011	559	Modification of description in *12.3.6 (1) PROTCMDm - Protection command register m (m = 2)*
		559	Modification of description in *Table 12-8 PROTCMDm register contents*
		574	Addition of description to *13.4.2 (2) OSTMnCNT - OSTM counter register*
		577	Addition of description to *13.4.2 (8) OSTMnCTL - OSTM control register*
		580	Modification of description in *Table 14-7 WDTA startup options*
		581	Modification of *Figure 14-1 Block diagram of WDTA*
		582	Addition of description to *14.4 Functional Description*
		585	Addition of description to *14.4.1 (3) WDTA settings after reset release*
		587	Addition of description to *14.4.3 Error detection*
		592	Modification of description in *14.5.2 (1) WDTAnWDTE - WDTA enable register*
		592	Modification of description in *Table 14-13 WDTAnRUN initial value*
		594	Modification of description in *14.5.2 (2) WDTAnEVAC - WDTA enable VAC register*
		594	Modification of description in *Table 14-16 WDTAnEVAC7 initial value*
		595	Modification of description in *14.5.2 (3) WDTAnREF - WDTA reference value register*
		596	Modification of description in *14.5.2 (4) WDTAnMD - WDTA mode register*
		597	Addition of description to *Table 14-19 WDTAnMD register contents*
		625	Modification of description in *15.8.4 (4) Simultaneous rewrite when INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.TAUAnRDCm that in turn is triggered by an external signal (method C2)*
		663	Deletion of description from and addition of description to *Table 15-25 TAUAnCMURm settings for delay count function*
		668	Deletion of description from and addition of description to *Table 15-29 TAUAnCMURm settings for one-pulse output function*
		679	Addition of description to *15.17.2 (1) Overview*
		683	Modification of description in *Table 15-42 Operating procedure for TAUAnTTINm input signal width measurement function*
		691	Addition of description to *15.17.4 (1) Overview*
		691	Modification of description in *15.17.4 (2) Equations*
		693	Addition of description to *Table 15-47 TAUAnCMORm settings for TAUAnTTINm input period count detection function*
		704	Modification of description in *Table 15-56 TAUAnCMURm settings for TAUAnTTINm input pulse interval judgment function*
		722	Modification of description in *Table 15-69 TAUAnCMORm settings for real-time output function type 2*
		722	Modification of description in *Table 15-70 TAUAnCMURm settings for real-time output function type 2*
		737	Modification of description in *Table 15-81 TAUAnCMURm settings for simultaneous rewrite trigger generation function type 2*
		764	Addition of description to *Figure 15-86 General timing diagram for TAUAnTTINm input position detection function*

Rev.	Date	Description	
		Page	Summary
0.02	Sep. 30, 2011	765	Addition of description to *Table 15-102 TAUAnCMORm settings for TAUAnTTINm input position detection function*
		786	Addition of description to *Table 15-117 TAUAnCMORm settings for the slave channel of the Trigger Start PWM output function*
		799	Modification of description in *Table 15-127 Control bit settings for slave channel 1 of the synchronous channel output mode 1*
		818	Modification of description in *15.24.2 (2) Equations*
		820	Addition of description to *Table 15-145 TAUAnCMORm settings for the master channel of the offset trigger output function*
		837	Modification of description in *15.25.2 (1) Overview*
		851	Deletion of description from *15.25.3 (1) Overview*
		935	Modification of description in *Table 15-254 TAUAnRDC register contents*
		996	Modification of description in *Table 16-23 TAUBnCMURm settings for one-pulse output function*
		1008	Addition of description to *16.15.2 (1) Overview*
		1012	Addition of description to *Table 16-36 Operating procedure for TAUBnTTINm input signal width measurement function*
		1021	Addition of description to *16.15.4 (1) Overview*
		1021	Modification of description in *16.15.4 (2) Equations*
		1023	Addition of description to *Table 16-41 TAUBnCMORm settings for TAUBnTTINm input period count detection function*
		1034	Addition of description to *Table 16-50 TAUBnCMURm settings for TAUBnTTINm input pulse interval judgment function*
		1064	Modification of description in *16.17.3 (2) Equations*
		1065	Addition of description to *Figure 16-73 General timing diagram for TAUBnTTINm input position detection function*
		1066	Addition of description to *Table 16-71 TAUBnCMORm settings for TAUBnTTINm input position detection function*
		1125	Modification of description in *16.21.2 (1) Overview*
		1148	Deletion of description from *16.21.3 (1) Overview*
		1156	Modification of description in *16.22.3 (3) TAUBnCMORm - TAUBnCMORm - TAUBn channel mode OS register*
		1168	Modification of description in *Table 16-151 TAUBnRDC register contents*
		1224	Addition of description to *17.16.2 (1) Overview*
		1228	Modification of description in *Table 17-32 Operating procedure for TAUJnTTINm input signal width measurement function*
		1235	Addition of description to *17.16.4 (1) Overview*
		1235	Modification of description in *17.16.4 (2) Equations*
		1237	Addition of description to *Table 17-37 TAUJnCMORm settings for TAUJnTTINm input period count detection function*
		1248	Addition of description to *Figure 17-49 General timing diagram for TAUJnTTINm input position detection function*

Rev.	Date	Description	
		Page	Summary
0.02	Sep. 30, 2011	1249	Addition of description to *Table 17-45 TAUJnCMORm settings for TAUJnTTINm input position detection function*
		1258	Deletion of description from *17.18.1 (4) (d) Simultaneous rewrite for the master channel*
		1293	Modification of description in *Table 18-12 Correction value examples*
		1302	Modification of description in *18.4.3 (1) RTCAnSUBC - RTCA sub-count register*
		1303	Modification of description in *18.4.3 (2) RTCAnSRBU - RTCA sub-count register read buffer*
		1305	Modification of description in *18.4.3 (4) RTCAnSCMP - RTCA sub-counter compare register*
		1319	Modification of description in *18.4.4 (13) RTCAnYEARC - RTCA year count register*
		1323	Modification of description in *18.4.5 (3) RTCAnCALC - RTCA calendar reading register*
		1374	Modification of description in *Table 20-1 Instances of UARTE*
		1376	Addition of description to *Table 20-6 URTEn I/O signals*
		1420	Modification of description in *Table 21-1 Instances of LMAn*
		1420	Modification of description in *Table 21-2 Channels of CNTAn*
		1431	Modification of description in *21.4.2 (3) Data transmission*
		1435	Deletion of description from *21.4.3 LIN master modes*
		1440	Modification of description in *21.4.3 (3) Data transmission*
		1447	Deletion of description from *21.4.4 Automatic checksum function*
		1450	Modification of description in *21.4.5 (2) Scheduler operation with automatic frame start function*
		1459	Deletion of description from *Table 21-23 LMAnSTRH register contents*
		1470	Modification of description in *Table 22-8 FCNn time stamp signals*
		1475	Addition of description to *22.3.2 Configuration*
		1483	Modification of description in *Table 22-16 FCN module register bit configuration*
		1485, 1486	Modification of description in *Table 22-17 Message buffer register bit configuration*
		1492	Modification of description in *22.6.1 (2) FCNnGMCSPRE - FCNn global clock selection register*
		1493	Modification of description in *22.6.1 (3) (a) FCNnGMABCTL read*
		1497	Modification of description in *22.6.2 (1) FCNnCMMKCTLaH - FCNn module mask control register*
		1499, 1500, 1501	Modification of description in and addition of description to *22.6.2 (2) (a) FCNnCMCLCTL read*
		1508	Modification of description in *22.6.2 (7) (b) FCNnCMISCTL write*
		1509	Modification of description in *22.6.2 (8)* FCNnCMBRPRS - FCNn module bit rate prescaler register
		1509	Modification of description in *Figure 22-3 FCN module clock*
		1510	Modification of description in *Figure 22-4 Data bit time*

Rev.	Date	Description	
		Page	Summary
0.02	Sep. 30, 2011	1510	Addition of description to *22.6.2 (9) FCNnCMBTCTL - FCNn module bit rate register*
		1519	Modification of description in *22.6.3 (1) FCNnMmDATxB/H/W, FCNn message data byte registers*
		1521	Modification of description in *22.6.3 (2) FCNnMmDTLGB - FCNn message data length register m*
		1522	Addition of description to *22.6.3 (3) FCNnMmSTRB - FCNn message configuration register m*
		1524	Modification of description in *22.6.3 (4) FCNnMmMID0H, FCNnMmMID1H, FCNnMmMID0W - FCNn message ID register m*
		1526	Modification of description in and deletion of description from *22.6.3 (5) FCNnMmCTL - FCNn message control register m*
		1532	Modification of figure in *Figure 22-6 Transition to operation modes*
		1532	Deletion of description from *22.7 CAN Controller Initialization*
		1534	Addition of description to *22.8.2 Receive data read*
		1534	Modification of figure in *Figure 22-7 FCNnMmCTL.FCNnMmDTNF and FCNnMmCTL.FCNnMmMUCF bit setting period (for standard ID format)*
		1548	Modification of description in *22.9.4 (2) Transmission abort process except for ABT transmission in normal operation mode with automatic block transmission (ABT)*
		1550	Addition of description to *22.10.1 (1) Entering FCN sleep mode*
		1562	Modification of description in *22.14.1 Baud rate setting conditions*
		1565	Modification of description in *22.14.2 Representative examples of baud rate settings*
		1565, 1566	Modification of title of *Table 22-23 Representative examples of baud rate settings (f_{CANPRE} = 8 MHz)*
		1567, 1568	Modification of title of*Table 22-24 Representative examples of baud rate settings (f_{CANPRE} = 16 MHz)*
		1571	Modification of *Figure 22-15 Re-initialization without Software Reset function*
		1571	Addition of description to *22.15.1 Initialization*
		1575	Modification of description in and deletion of figure from *Figure 22-19 Message buffer redefinition during transmission*
		1576	Modification of description in *Figure 22-20 Message transmit processing*
		1578	Modification of description in *Figure 22-22 Transmission via interrupt (using FCNnCMLOSTR register)*
		1579	Modification of description in *Figure 22-23 Transmission via interrupt (using FCNnCMTGTX register)*
		1581	Modification of description in *Figure 22-24 Transmission via software polling*
		1583	Modification of *Figure 22-26 Processing to abort transmission other than ABT transmission (in normal operation mode with ABT)* and its title
		1584	Addition of *Figure 22-27 ABT transmission request abort processing (normal operation mode with ABT) (1)*
		1584	Addition of description to *Figure 22-27 ABT transmission request abort processing (normal operation mode with ABT) (1)*

Rev.	Date	Description	
		Page	Summary
0.02	Sep. 30, 2011	1585	Addition of figure to *Figure 22-28 ABT transmission request abort processing (normal operation mode with ABT) (2)*
		1585	Addition of description to *Figure 22-28 ABT transmission request abort processing (normal operation mode with ABT) (2)*
		1586	Deletion of figure from and modification of description in *Figure 22-29 ABT transmission request abort processing (normal operation mode with ABT) with transmit completion flag*
		1586	Addition of description to *Figure 22-29 ABT transmission request abort processing (normal operation mode with ABT) with transmit completion flag*
		1587	Deletion of description from *22.15.2 Message transmission*
		1591	Addition of description to *22.15.3 Message reception*
		1592	Deletion of description from *22.15.3 Message reception*
		1593	Modification of *Figure 22-35 Reception via software polling*
		1594	Modification of description in *Figure 22-36 Setting FCN sleep mode/stop mode*
		1596	Modification of description in *Figure 22-38 Bus-off recovery* and its title
		1598	Modification of description in *Figure 22-41 Error handling*
		1600	Modification of description in and addition of description to *Figure 22-43 Setting CPU standby (from FCN stop mode)*
		1612	Modification of description in *Table 23-8 MAC control register list*
		1712	Modification of description in *Table 23-94 ETHAnTXABTCNT register contents*
		1730	Modification of description in *23.4.5 (5) ETHAnCDMACM - Transmit checksum DMA control mode setting register*
		1790	Addition of description to *Table 24-7 CSIGn I/O signals*
		1813	Modification of description in *Table 24-10 CSIGn register overview*
		1821	Modification of description in *Table 24-18 CSIGnBCTL0 register contents*
		1826	Modification of description in *24.4 (10) CSIGnRX0 - Reception data register 0*
		1838	Addition of description to *Table 25-8 CSIHn I/O signals*
		2081	Addition of description to *Figure 26-16 Master operate setting sequence during continuous transfer mode (single master environment)*
		2099	Deletion of description from *27.1.2 (2) Baudrate generators*
		2100	Modification of description in *Table 27-9 CLKDakDIV register contents*
		2115	Addition of description to *Table 27-13 IISAnCTL register contents*
		2120	Addition of description to *Table 27-16 IISAnSTC register contents*
		2122	Modification of description in *27.4.4 (2) IISAnTX - IISA transmission data register*
		2189	Modification of description in *28.6.2 (13) PCMnCFG - PCM signal configuration register*
		2189	Deletion of description from *Table 28-24 PCMnCFG register contents*
		2329	Modification of description in *Table 32-2 ADCAn channel indices*
		2331	Modification of description in *Table 32-7 ADCAn I/O signals*
		2347	Modification of *Figure 32-2 Block diagram of the ADCAn*

Rev.	Date	Description	
		Page	Summary
0.02	Sep. 30, 2011	2347	Deletion of description from *Table 32-9 Sampling and conversion times (product dependent)*
		2348	Deletion of description from *32.3.9 Resolution, sampling and conversion times*
		2362	Addition of description to *32.3.14 (1) Channel S/H function*
		2377	Modification of description in *Table 32-16 ADCAnCGi register contents*
		2393	Modification of description in *Table 32-32 ADCAnCTL2 register contents*
		2401	Deletion of figure from *32.5.4 Notes on application design*
		2404	Modification of *Figure 32-23 Wiring example*
		2417	Modification of description in *33.3 (1) EPC - Emulation break control register*
		2418	Modification of description in *33.4 Connection with On-Chip Debug Emulator*
		2421	Modification of description in *Table 34-2 Power supply pins*
		2422	Modification of description in *Figure 34-1 Power supply scheme*
		2430	Addition of description to *Table 34-3 Power supplies*

V850E2/Sx4-H Hardware User's Manual

Publication Date: Rev. 0.02 September 30, 2011

Published by: Renesas Electronics Corporation

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.1

V850E2/Sx4-H

Renesas Electronics Corporation

R01UH0107EJ0002