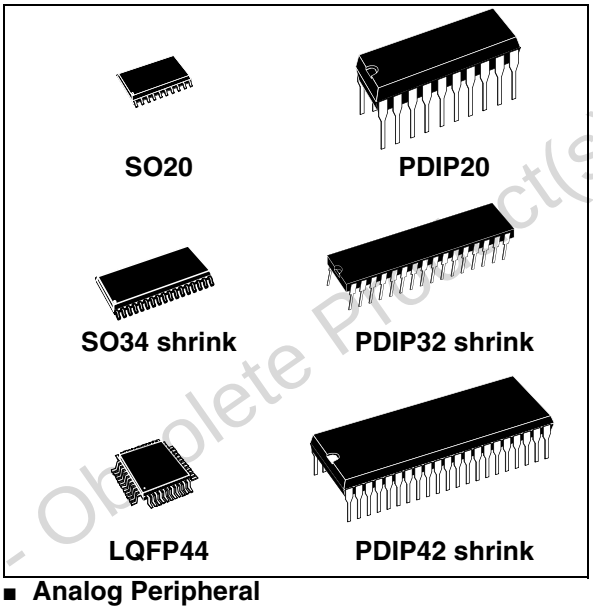
ST7262xxx

Low speed USB 8-bit MCU with 3 endpoints, Flash or ROM memory, LVD, WDG, 10-bit ADC, 2 timers, SCI, SPI

Features

- **Memories**
  - 8 or 16 Kbyte Program memory (ROM or Dual voltage FLASH) with read-write protection
  - In-Application and In-Circuit Programming for FLASH versions
  - 384 to 768 bytes RAM (128-byte stack)
- **Clock, Reset and Supply Management**
  - Enhanced Reset System (Power On Reset)
  - Low Voltage Detector (LVD)
  - Clock-out capability
  - 6 or 12 MHz Oscillator (8, 4, 2, 1 MHz internal frequencies)
  - 3 Power saving modes
- **USB (Universal Serial Bus) Interface**
  - DMA for low speed applications compliant with USB specification (version 2.0):
  - Integrated 3.3V voltage regulator and transceivers
  - Suspend and Resume operations
  - 3 Endpoints
- **Up to 31 I/O Ports**
  - Up to 31 multifunctional bidirectional I/O lines
  - Up to 12 External interrupts (3 vectors)
  - 13 alternate function lines
  - 8 high sink outputs (8 mA@0.4 V/20 mA@1.3 V)
  - 2 true open drain pins (N buffer 8 mA@0.4 V)
- **3 Timers**
  - Configurable watchdog timer (8 to 500 ms timeout)
  - 8-bit Auto Reload Timer (ART) with 2 Input Captures, 2 PWM outputs and External Clock
  - 8-bit Time Base Unit (TBU) for generating periodic interrupts cascadable with ART



- **Analog Peripheral**
  - 10-bit A/D Converter with up to 8 input pins.
- **2 Communications Interfaces**
  - Asynchronous Serial Communication interface
  - Synchronous Serial Peripheral Interface
- **Instruction Set**
  - 8-bit data manipulation
  - 63 basic instructions
  - 17 main addressing modes
  - 8 x 8 unsigned multiply instruction
  - True bit manipulation
- **Nested interrupts**
- **Development Tools**
  - Full hardware/software development package

Device Summary

Features	ST72623F2	ST72621K4	ST72622L2	ST72621L4	ST72621J4
Program memory - Kbytes	8	16	8	16	16
RAM (stack) - bytes	384 (128)	768 (128)	384 (128)	768 (128)	768 (128)
Peripherals	USB, Watchdog, Low Voltage Detector, 8-bit Auto-Reload timer, Timebase unit, A/D Converter				
Serial I/O	-	SPI + SCI	SPI	SPI + SCI	
I/Os	11	21	23		31
Operating Supply	4.0V to 5.5V (Low voltage 3.0V to 5.5V ROM versions available)				
Operating Temperature	0°C to +70°C				
Packages	PDIP20/SO20	PDIP32	SO34		PDIP42/LQFP44

Table of Contents

1 INTRODUCTION

The ST7262 and ST72F62 devices are members of the ST7 microcontroller family designed for USB applications.

All devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

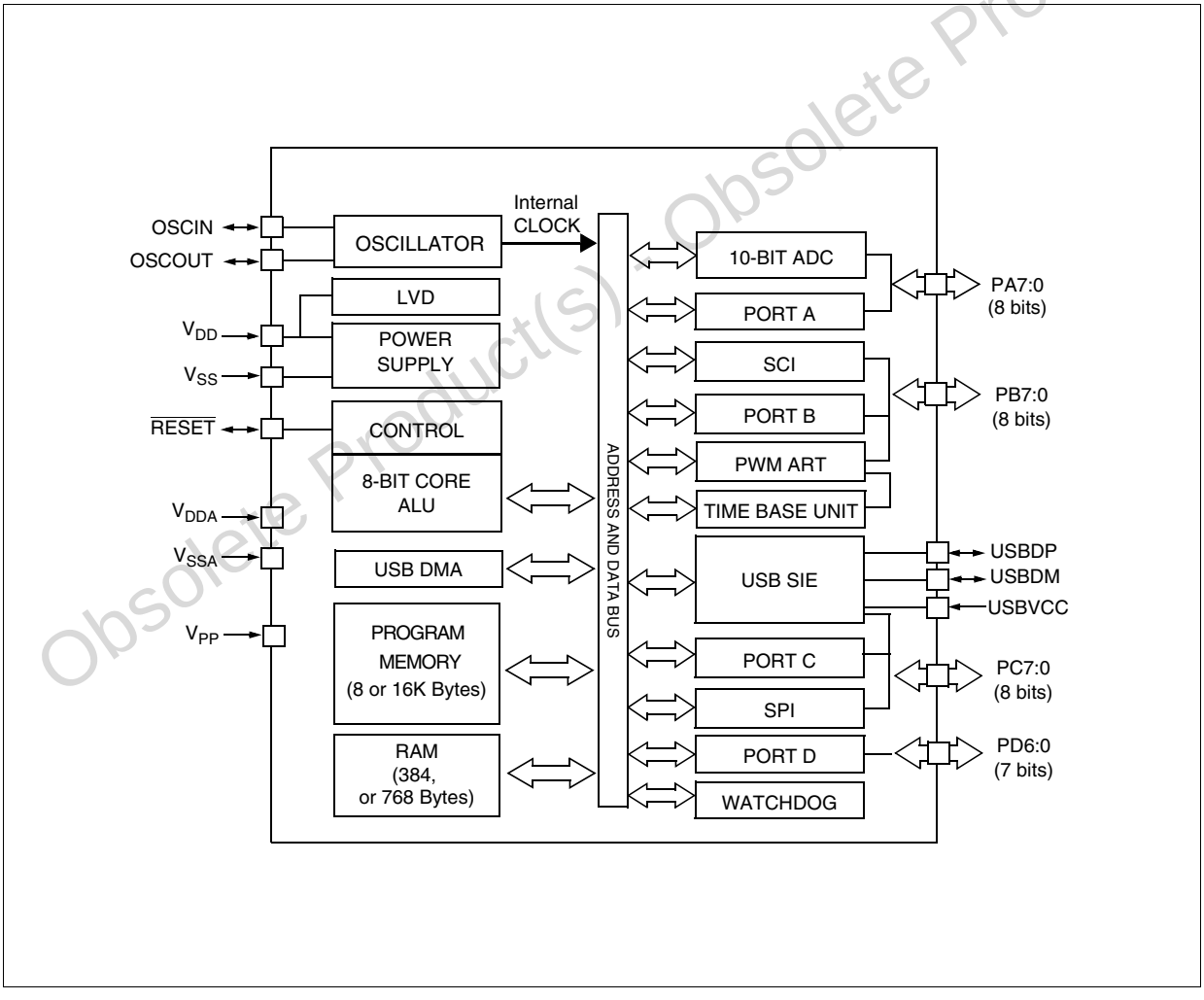
The ST7262 devices are ROM versions.

The ST72F62 versions feature dual-voltage FLASH memory with FLASH Programming capability.

Under software control, all devices can be placed in WAIT, SLOW, or HALT mode, reducing power consumption when the application is in idle or standby state.

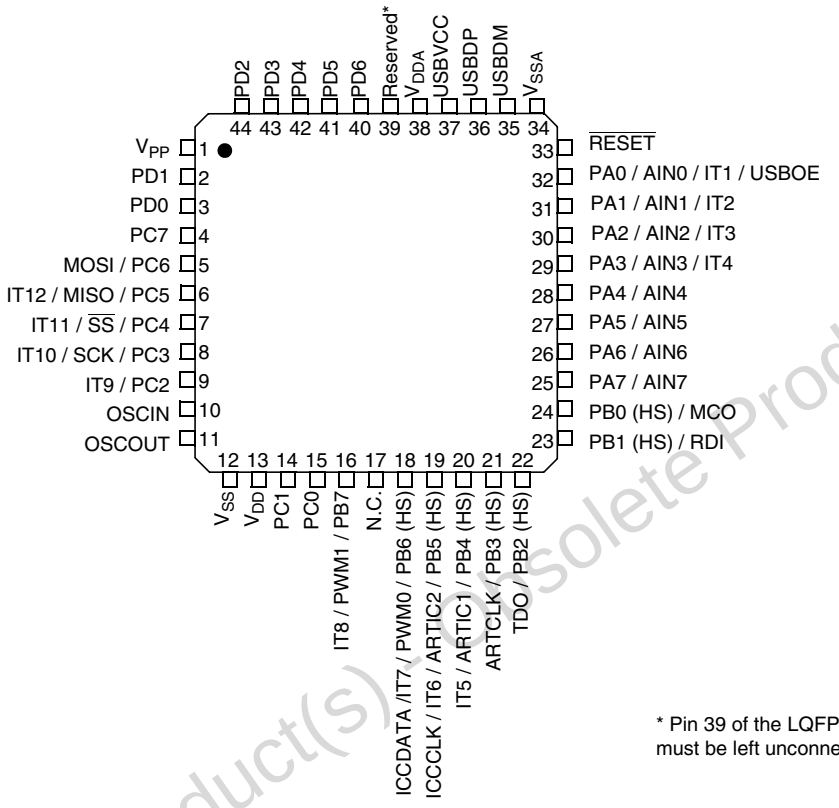
The enhanced instruction set and addressing modes of the ST7 offer both power and flexibility to software developers, enabling the design of highly efficient and compact application code. In addition to standard 8-bit data management, all ST7 microcontrollers feature true bit manipulation, 8x8 unsigned multiplication and indirect addressing modes.

Figure 1. General Block Diagram

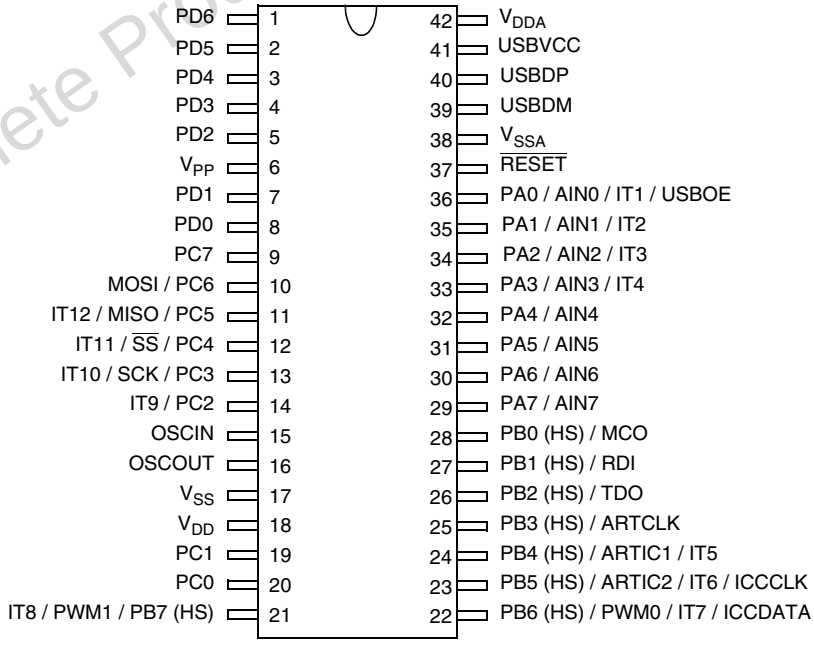


2 PIN DESCRIPTION

Figure 2. 44-pin LQFP and 42-Pin SDIP Package Pinouts



* Pin 39 of the LQFP44 package must be left unconnected.



PIN DESCRIPTION (Cont'd)

Figure 3. 34-Pin SO and 32-Pin SDIP Package Pinouts

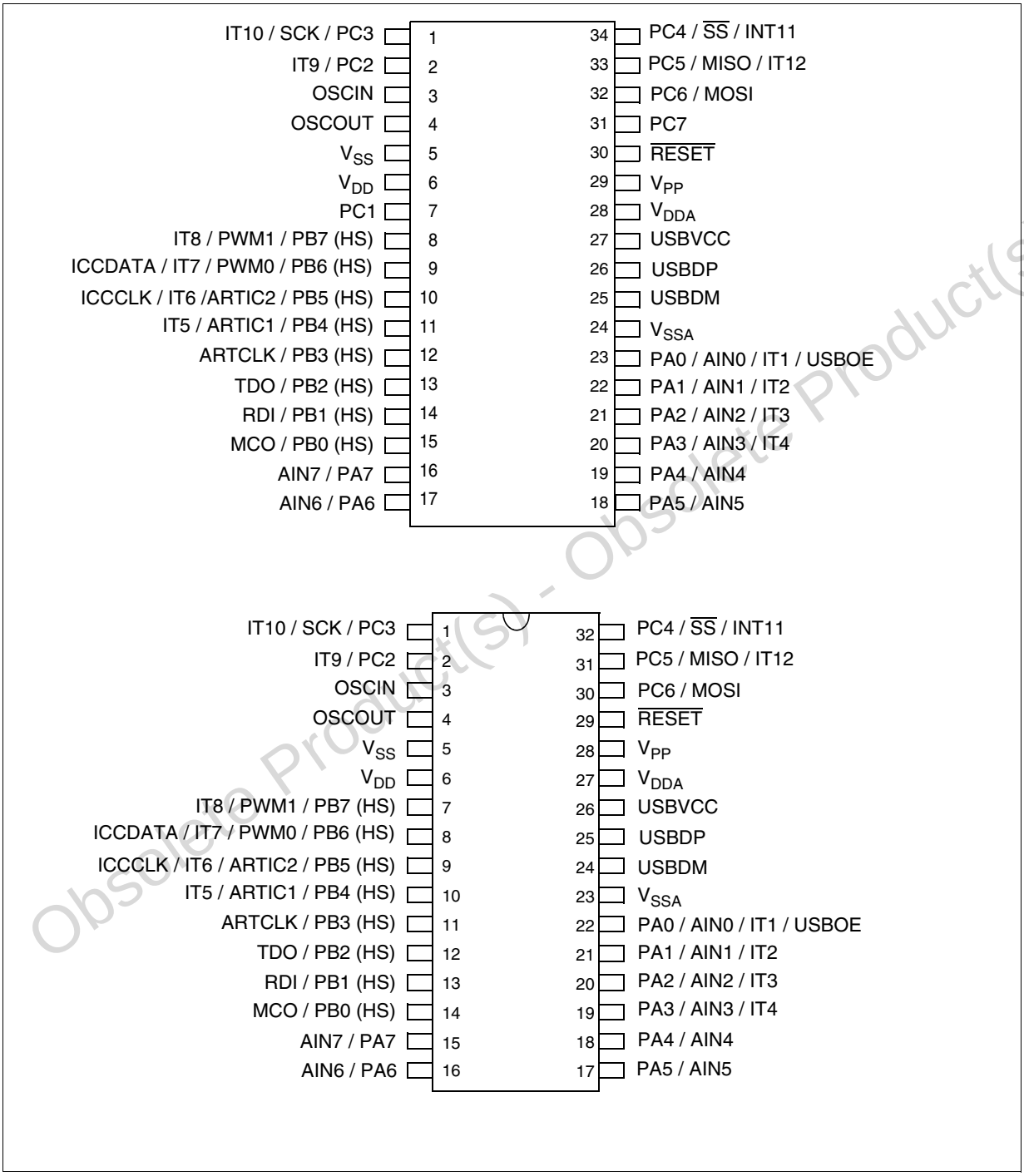


Figure 4. 20-pin SO20 Package Pinout

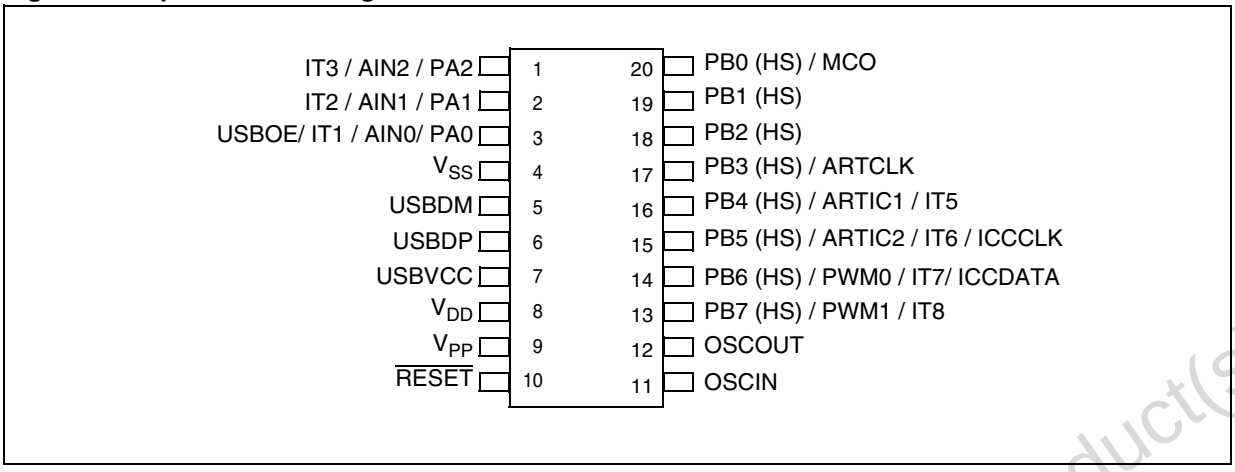
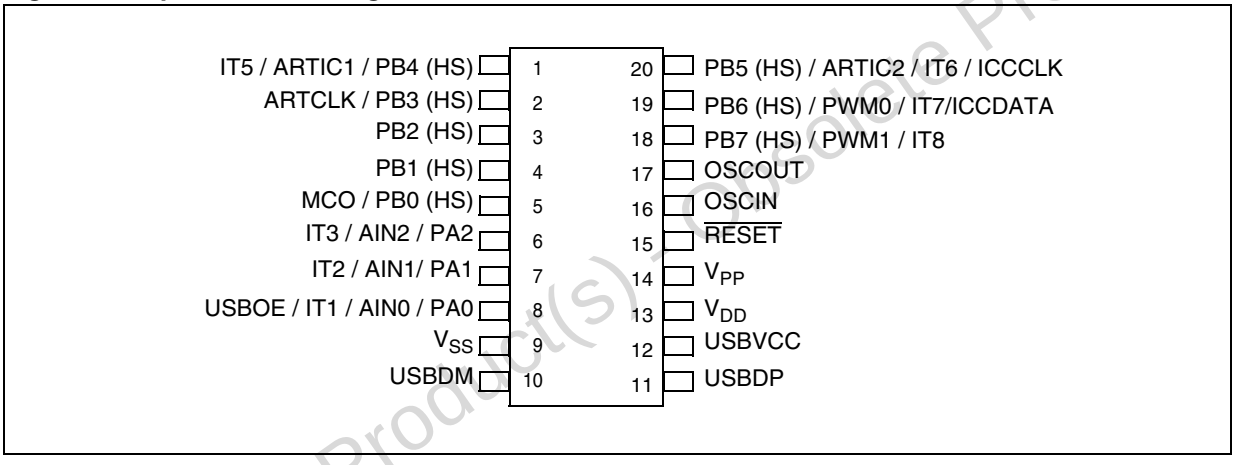


Figure 5. 20-pin DIP20 Package Pinout



PIN DESCRIPTION (Cont'd)

Legend / Abbreviations:

Type: I = Input, O = Output, S = Supply

Input level: A = Dedicated analog input

Input level: C = CMOS $0.3V_{DD}/0.7V_{DD}$,
C_T= CMOS $0.3V_{DD}/0.7V_{DD}$ with input trigger

Output level: HS = High Sink (on N-buffer only)

Port configuration capabilities:

– Input:float = floating, wpu = weak pull-up, int = interrupt (\ =falling edge, / =rising edge), ana = analog

– Output: OD = open drain, T = true open drain (N buffer 8mA@0.4 V), PP = push-pull

Table 1. Device Pin Description

Pin n°						Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
										Input				Output			
LQFP44	DIP42	SO34	DIP32	SO20	DIP20			Input	Output	float	wpu	int	ana	OD	PP		
1	6	29	28	9	14	V_{PP}	S				x					FLASH programming voltage (12V), must be tied low in user mode.	
2	7	-	-	-	-	PD1	I/O	C_T			x				x	**Port D1**	
3	8	-	-	-	-	PD0	I/O	C_T			x				x	**Port D0**	
4	9	31	-	-	-	PC7	I/O	C_T			x				x	**Port C7**	
5	10	32	30	-	-	PC6/MOSI	I/O	C_T			x				x	**Port C6**	SPI Master Out / Slave In [1]
6	11	33	31	-	-	PC5/MISO/IT12	I/O	C_T			x	x			x	**Port C5**	SPI Master In / Slave Out [1] / Interrupt 12 input
7	12	34	32	-	-	PC4/SS/IT11	I/O	C_T			x	x			x	**Port C4**	SPI Slave Select (active low) [1]/ Interrupt 11 input
8	13	1	1	-	-	PC3/SCK/IT10	I/O	C_T			x	x			x	**Port C3**	SPI Serial Clock [1]/ Interrupt 10 input
9	14	2	2	-	-	PC2/IT9	I/O	C_T			x	x			x	**Port C2**	Interrupt 9 input
10	15	3	3	11	16	OSCIN										These pins are used connect an external clock source to the on-chip main oscillator.	
11	16	4	4	12	17	OSCOUT											
12	17	5	5	4	9	V_{SS}	S									Digital Ground Voltage	
13	18	6	6	8	13	V_{DD}	S									Digital Main Power Supply Voltage	
14	19	7	-	-	-	PC1	I/O	C_T		x				T		**Port C1**	
15	20	-	-	-	-	PC0	I/O	C_T		x				T		**Port C0**	
16	21	8	7	13	18	PB7/PWM1/IT8/ RX_SEZ/DA-TAOUT/DA9	I/O	C_T	HS	x		\			x	**Port B7**	ART PWM output 1/ Interrupt 8 input
17	-	-				N.C.										Not Connected	

Pin n°						Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
										Input				Output			
LQFP44	DIP42	SO34	DIP32	SO20	DIP20			Input	Output	float	wpu	int	ana	OD	PP		
18	22	9	8	14	19	PB6/PWM0/IT7/ ICCDATA	I/O	C_T	HS	x		\			x	**Port B6**	ART PWM output 0/ Interrupt 7 input/In-Circuit Communication Data
19	23	10	9	15	20	PB5/ARTIC2/IT6/ ICCCLK	I/O	C_T	HS	x		/			x	**Port B5**	ART Input Capture 2/ Interrupt 6 input/ In-Circuit Communication Clock
20	24	11	10	16	1	PB4/ARTIC1/IT5	I/O	C_T	HS	x		/			x	**Port B4**	ART Input Capture 1/Interrupt 5 input
21	25	12	11	17	2	PB3/ARTCLK	I/O	C_T	HS	x					x	**Port B3**	ART Clock input
22	26	13	12	18	3	PB2/TDO	I/O	C_T	HS	x					x	**Port B2**	SCI Transmit Data Output [1]
23	27	14	13	19	4	PB1/RDI	I/O	C_T	HS	x					x	**Port B1**	SCI Receive Data Input [1]
24	28	15	14	20	5	PB0/MCO	I/O	C_T	HS	x					x	**Port B0**	CPU clock output
25	29	16	15	-	-	PA7/AIN7	I/O	C_T		x			x		x	**Port A7**	ADC Analog Input 7
26	30	17	16	-	-	PA6/AIN6	I/O	C_T		x			x		x	**Port A6**	ADC Analog Input 6
27	31	18	17	-	-	PA5/AIN5	I/O	C_T		x			x		x	**Port A5**	ADC Analog Input 5
28	32	19	18	-	-	PA4/AIN4	I/O	C_T		x			x		x	**Port A4**	ADC Analog Input 4
29	33	20	19	-	-	PA3/AIN3/IT4	I/O	C_T		x		\	x		x	**Port A3**	ADC Analog Input 3/ Interrupt 4 input
30	34	21	20	1	6	PA2/AIN2/IT3	I/O	C_T		x		\	x		x	**Port A2**	ADC Analog Input 2/ Interrupt 3 input
31	35	22	21	2	7	PA1/AIN1/IT2	I/O	C_T		x		\	x		x	**Port A1**	ADC Analog Input 1/ Interrupt 2 input
32	36	23	22	3	8	PA0/AIN0/IT1/ USBOE	I/O	C_T		x		\	x		x	**Port A0**	ADC Analog Input 0/ Interrupt 1 input/ USB Output Enable
33	37	30	29	10	15	$\overline{\text{RESET}}$	I/O	C								Top priority non maskable interrupt (active low)	
34	38	24	23	-	-	V_{SSA}	S									Analog Ground Voltage, must be connected externally to V_{SS}.	
35	39	25	24	5	10	USBDM	I/O									USB bidirectional data (data -)	
36	40	26	25	6	11	USBDP	I/O									USB bidirectional data (data +)	
37	41	27	26	7	12	USBVCC	S									USB power supply 3.3V output	
38	42	28	27	-	-	V_{DDA}	S									Analog Power Supply Voltage, must be connected externally to V_{DD}.	
39	-	-	-	-	-	Reserved										Must be left unconnected.	
40	1	-	-	-	-	PD6	I/O	C_T			x				x	**Port D6**	
41	2	-	-	-	-	PD5	I/O	C_T			x				x	**Port D5**	
42	3	-	-	-	-	PD4	I/O	C_T			x				x	**Port D4**	

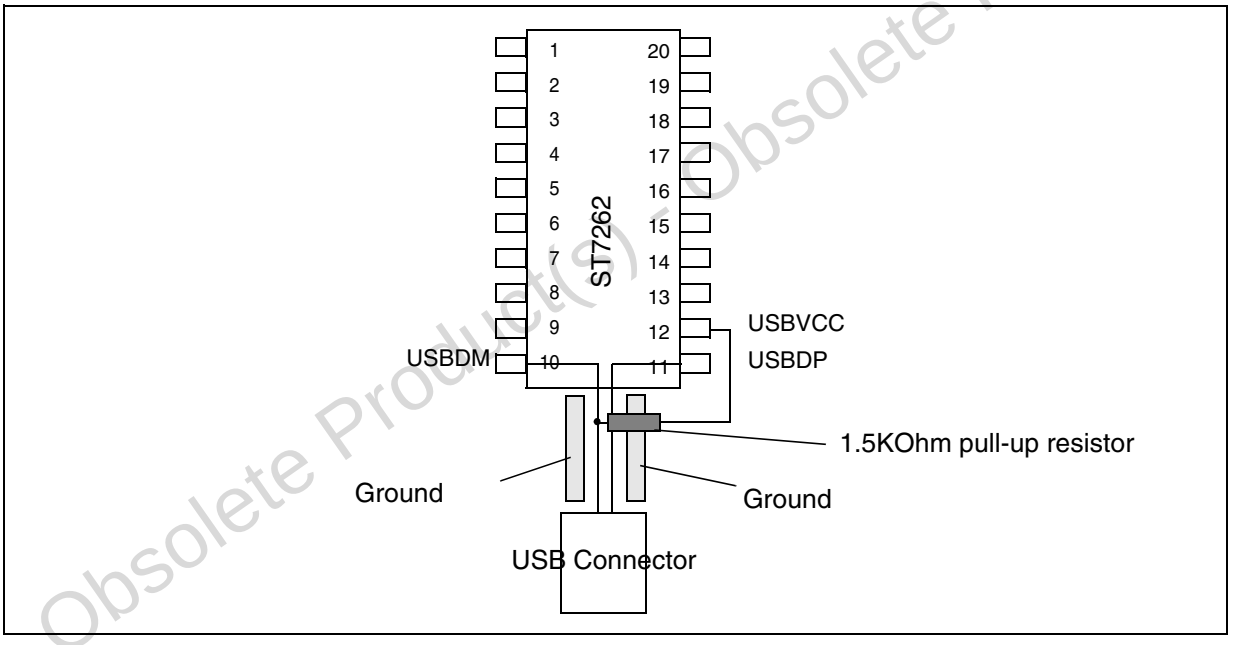
Pin n°						Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
										Input				Output			
LQFP44	DIP42	SO34	DIP32	SO20	DIP20			Input	Output	float	wpu	int	ana	OD	PP		
43	4	-	-	-	-	PD3	I/O	C_T			x				x	**Port D3**	
44	5	-	-	-	-	PD2	I/O	C_T			x				x	**Port D2**	

Note 1: Peripheral not present on all devices. Refer to "Device Summary" on page 1.

2.1 PCB LAYOUT RECOMMENDATION

In the case of DIP20 devices the user should lay-out the PCB so that the DIP20 ST7262 device and the USB connector are centered on the same axis ensuring that the D- and D+ lines are of equal length. Refer to Figure 6

Figure 6. Recommended PCB Layout for USB Interface with DIP20 package



3 REGISTER & MEMORY MAP

As shown in the Figure 7, the MCU is capable of addressing 64K bytes of memories and I/O registers.

The available memory locations consist of 64 bytes of register locations, 768 bytes of RAM and up to 16 Kbytes of user program memory. The RAM space includes up to 128 bytes for the stack from 0100h to 017Fh.

The highest address bytes contain the user reset and interrupt vectors.

IMPORTANT: Memory locations marked as “Reserved” must never be accessed. Accessing a reserved area can have unpredictable effects on the device.

Figure 7. Memory Map

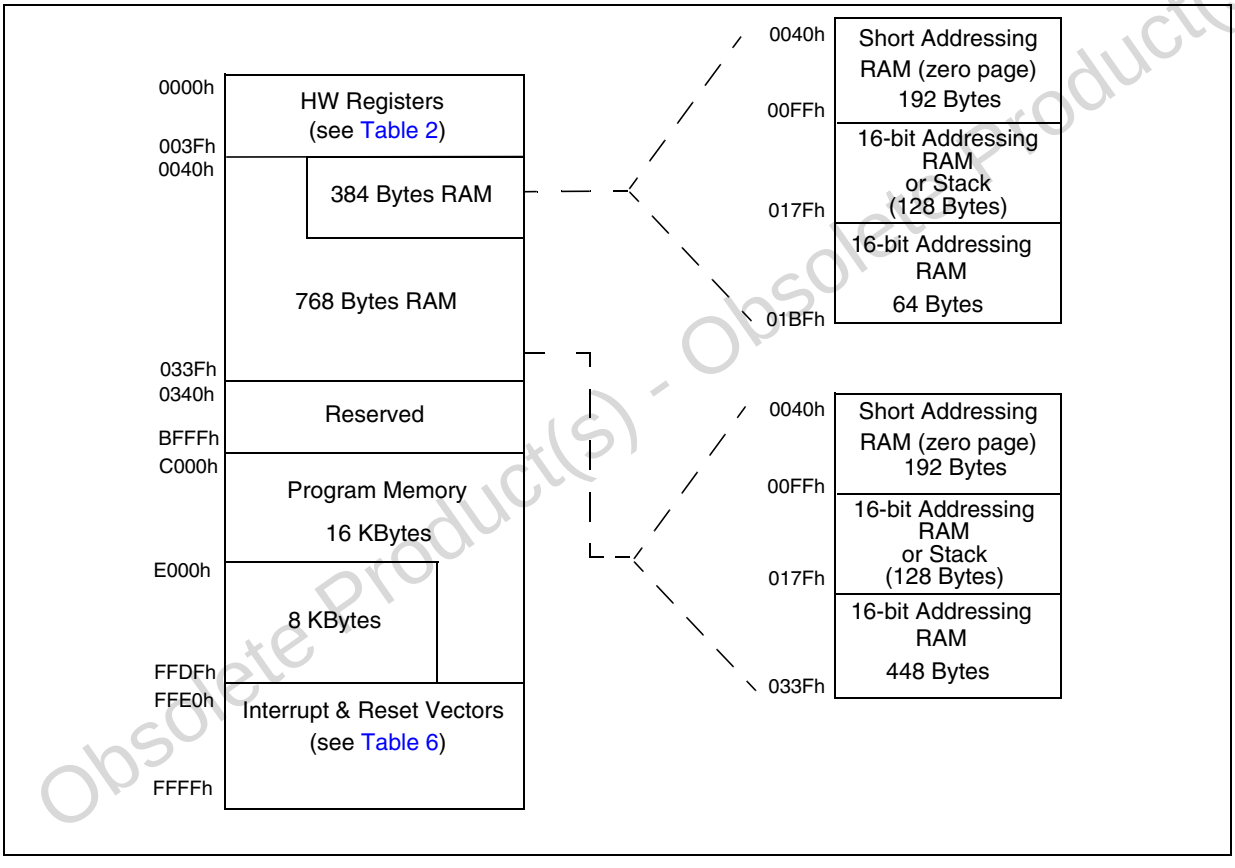


Table 2. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0000h 0001h	Port A	PADR PADDR	Port A Data Register Port A Data Direction Register	00h[1)] 00h	R/W[2)] R/W[2)]
0002h 0003h	Port B	PBDR PBDDR	Port B Data Register Port B Data Direction Register	00h[1)] 00h	R/W[2)] R/W[2)]
0004h 0005h	Port C	PCDR PCDDR	Port C Data Register Port C Data Direction Register	00h[1)] 00h	R/W[2)] R/W[2)]
0006h 0007h	Port D	PDDR PDDDR	Port D Data Register Port D Data Direction Register	00h[1)] 00h	R/W[2)] R/W[2)]
0008h		ITRFRE1	Interrupt Register 1	00h	R/W
0009h		MISC	Miscellaneous Register	00h	R/W
000Ah 000Bh 000Ch	ADC	ADCDRMSB ADCDRLSB ADCCSR	ADC Data Register (bit 9:2) ADC Data Register (bit 1:0) ADC Control Status Register	00h 00h 00h	Read Only Read Only R/W
000Dh	WDG	WDGCR	Watchdog Control Register	7Fh	R/W
000Eh 0010h	Reserved Area (3 Bytes)				
0011h 0012h 0013h	SPI	SPIDR SPICR SPICSR	SPI Data I/O Register SPI Control Register SPI Control Status Register	xxh 0xh 00h	R/W R/W Read Only
0014h 0015h 0016h 0017h 0018h 0019h 001Ah 001Bh 001Ch	PWM ART	PWMDCR1 PWMDCR0 PWMCR ARTCSR ARTCAR ARTARR ARTICCSR ARTICR1 ARTICR2	PWM AR Timer Duty Cycle Register 1 PWM AR Timer Duty Cycle Register 0 PWM AR Timer Control Register Auto-Reload Timer Control/Status Register Auto-Reload Timer Counter Access Register Auto-Reload Timer Auto-Reload Register ART Input Capture Control/Status Register ART Input Capture Register 1 ART Input Capture Register 2	00h 00h 00h 00h 00h 00h 00h 00h 00h	R/W R/W R/W R/W R/W R/W R/W Read Only Read Only
001Dh 001Eh 001Fh 0020h 0021h 0022h 0023h 0024h	SCI	SCIERPR SCIETPR SCISR SCIDR SCIBRR SCICR1 SCICR2	SCI Extended Receive Prescaler register SCI Extended Transmit Prescaler Register Reserved Area SCI Status register SCI Data register SCI Baud Rate Register SCI Control Register 1 SCI Control Register 2	00h 00h -- C0h xxh 00h x000 0000b 00h	R/W R/W Read Only R/W R/W R/W R/W

Address	Block	Register Label	Register Name	Reset Status	Remarks
0025h	USB	USBPIDR	USB PID Register	x0h	Read Only
0026h		USBDMAR	USB DMA Address register	xxh	R/W
0027h		USBIDR	USB Interrupt/DMA Register	x0h	R/W
0028h		USBISTR	USB Interrupt Status Register	00h	R/W
0029h		USBIMR	USB Interrupt Mask Register	00h	R/W
002Ah		USBCTLR	USB Control Register	06h	R/W
002Bh		USBDADDR	USB Device Address Register	00h	R/W
002Ch		USBEP0RA	USB Endpoint 0 Register A	0000 xxxxb	R/W
002Dh		USBEP0RB	USB Endpoint 0 Register B	80h	R/W
002Eh		USBEP1RA	USB Endpoint 1 Register A	0000 xxxxb	R/W
002Fh		USBEP1RB	USB Endpoint 1 Register B	0000 xxxxb	R/W
0030h		USBEP2RA	USB Endpoint 2 Register A	0000 xxxxb	R/W
0031h		USBEP2RB	USB Endpoint 2 Register B	0000 xxxxb	R/W
0032h to 0035h	Reserved Area (4 Bytes)				
0036h	TBU	TBUCV	TBU Counter Value Register	00h	R/W
0037h		TBUCSR	TBU Control/Status Register	00h	R/W
0038h	FLASH	FCSR	Flash Control/Status Register	00h	R/W
0039h		ITRFRE2	Interrupt Register 2	00h	R/W
003Ah to 003Fh	Reserved Area (6 Bytes)				

Legend: x=undefined, R/W=read/write

Notes:

1. The contents of the I/O port DR registers are readable only in output configuration. In input configuration, the values of the I/O pins are returned instead of the DR register contents.
2. The bits associated with unavailable pins must always be kept at their reset value.

4 FLASH PROGRAM MEMORY

4.1 INTRODUCTION

The ST7 dual voltage High Density Flash (HDFlash) is a non-volatile memory that can be electrically erased as a single block or by individual sectors and programmed on a Byte-by-Byte basis using an external V_{PP} supply.

The HDFlash devices can be programmed and erased off-board (plugged in a programming tool) or on-board using ICP (In-Circuit Programming) or IAP (In-Application Programming).

The array matrix organisation allows each sector to be erased and reprogrammed without affecting other sectors.

4.2 MAIN FEATURES

- 3 Flash programming modes:
  - Insertion in a programming tool. In this mode, all sectors including option bytes can be programmed or erased.
  - ICP (In-Circuit Programming). In this mode, all sectors including option bytes can be programmed or erased without removing the device from the application board.
  - IAP (In-Application Programming) In this mode, all sectors except Sector 0, can be programmed or erased without removing the device from the application board and while the application is running.
- ICT (In-Circuit Testing) for downloading and executing user application test patterns in RAM
- Read-out protection
- Register Access Security System (RASS) to prevent accidental programming or erasing

4.3 STRUCTURE

The Flash memory is organised in sectors and can be used for both code and data storage.

Depending on the overall Flash memory size in the microcontroller device, there are up to three user sectors (see Table 3). Each of these sectors can be erased independently to avoid unnecessary erasing of the whole Flash memory when only a partial erasing is required.

The first two sectors have a fixed size of 4 Kbytes (see Figure 8). They are mapped in the upper part of the ST7 addressing space so the reset and interrupt vectors are located in Sector 0 (F000h-FFFFh).

Table 3. Sectors available in Flash devices

Flash Size (bytes)	Available Sectors
4K	Sector 0
8K	Sectors 0,1
> 8K	Sectors 0,1, 2

4.3.1 Read-out Protection

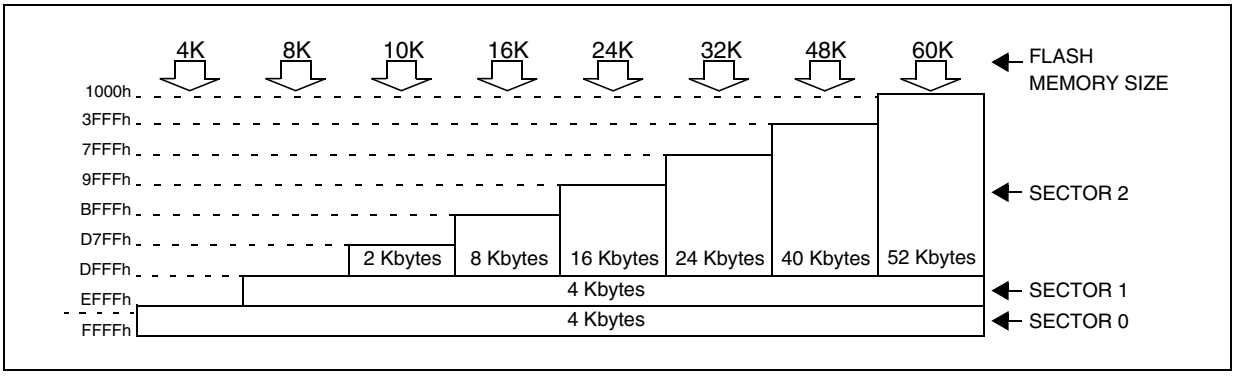
Read-out protection, when selected, provides a protection against Program Memory content extraction and against write access to Flash memory. Even if no protection can be considered as totally unbreakable, the feature provides a very high level of protection for a general purpose microcontroller.

In Flash devices, this protection is removed by reprogramming the option. In this case, the entire program memory is first automatically erased and the device can be reprogrammed.

Read-out protection selection depends on the device type:

- In Flash devices it is enabled and removed through the FMP_R bit in the option byte.
- In ROM devices it is enabled by mask option specified in the Option List.

Figure 8. Memory Map and Sector Address



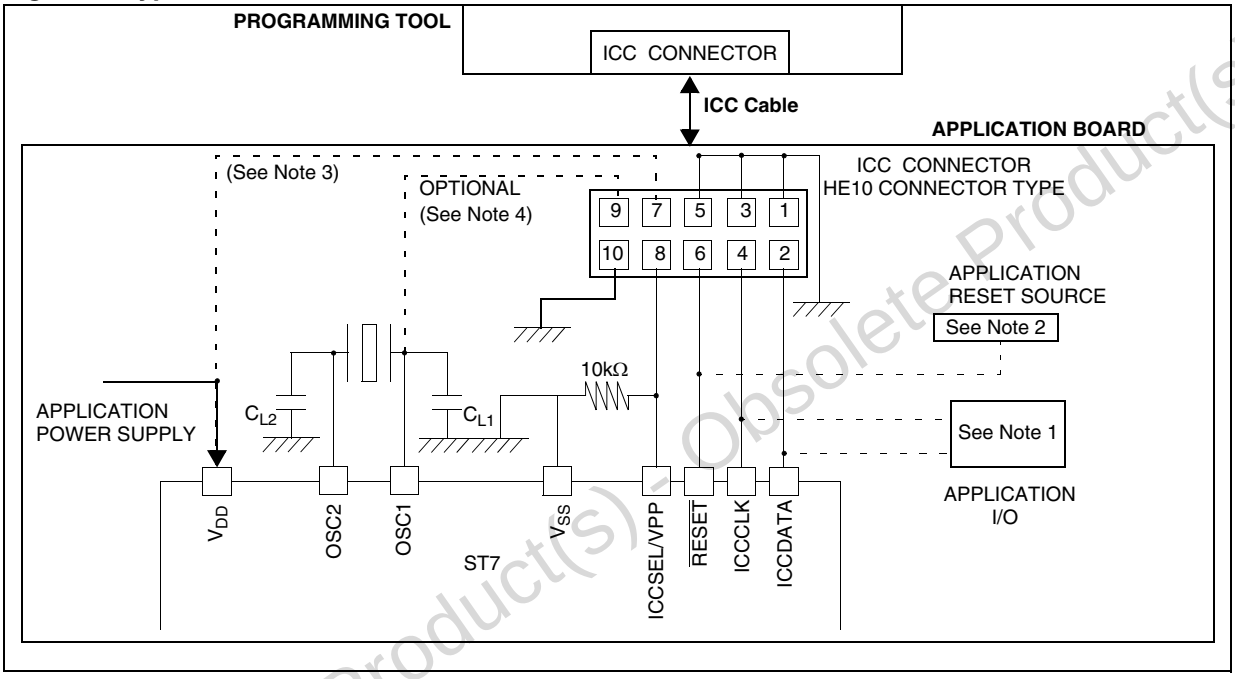
FLASH PROGRAM MEMORY (Cont'd)

4.4 ICC INTERFACE

ICC (In-Circuit Communication) needs a minimum of four and up to six pins to be connected to the programming tool (see Figure 9). These pins are:

- $\overline{\text{RESET}}$: device reset
- V_{SS}: device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input/output serial data pin
- ICCSEL/V_{PP}: programming voltage
- OSC1(or OSCIN): main clock input for external source (optional)
- V_{DD}: application board power supply (see Figure 9, Note 3)

Figure 9. Typical ICC Interface



Notes:

1. If the ICCCLK or ICCDATA pins are only used as outputs in the application, no signal isolation is necessary. As soon as the Programming Tool is plugged to the board, even if an ICC session is not in progress, the ICCCLK and ICCDATA pins are not available for the application. If they are used as inputs by the application, isolation such as a serial resistor has to implemented in case another device forces the signal. Refer to the Programming Tool documentation for recommended resistor values.

2. During the ICC session, the programming tool must control the $\overline{\text{RESET}}$ pin. This can lead to conflicts between the programming tool and the application reset circuit if it drives more than 5mA at high level (push pull output or pull-up resistor<1K). A schottky diode can be used to isolate the application RESET circuit in this case. When using a classical RC network with R > 1K or a reset management IC with open drain output and pull-up resistor > 1K, no additional components are needed. In all cases the user must ensure that no external reset is generated by the application during the ICC session.

3. The use of Pin 7 of the ICC connector depends on the Programming Tool architecture. This pin must be connected when using most ST Programming Tools (it is used to monitor the application power supply). Please refer to the Programming Tool manual.

4. Pin 9 has to be connected to the OSC1 or OSCIN pin of the ST7 when the clock is not available in the application or if the selected clock option is not programmed in the option byte. ST7 devices with multioscillator capability need to have OSC2 grounded in this case.

FLASH PROGRAM MEMORY (Cont'd)

4.5 ICP (IN-CIRCUIT PROGRAMMING)

To perform ICP the microcontroller must be switched to ICC (In-Circuit Communication) mode by an external controller or programming tool.

Depending on the ICP code downloaded in RAM, Flash memory programming can be fully customized (number of bytes to program, program locations, or selection serial communication interface for downloading).

When using an STMicroelectronics or third-party programming tool that supports ICP and the specific microcontroller device, the user needs only to implement the ICP hardware interface on the application board (see Figure 9). For more details on the pin locations, refer to the device pinout description.

4.6 IAP (IN-APPLICATION PROGRAMMING)

This mode uses a BootLoader program previously stored in Sector 0 by the user (in ICP mode or by plugging the device in a programming tool).

This mode is fully controlled by user software. This allows it to be adapted to the user application, (user-defined strategy for entering programming mode, choice of communications protocol used to fetch the data to be stored, etc.). For example, it is possible to download code from the SPI, SCI or other type of serial interface and program it in the Flash. IAP mode can be used to program any of the Flash sectors except Sector 0, which is write/erase protected to allow recovery in case errors occur during the programming operation.

4.7 RELATED DOCUMENTATION

For details on Flash programming and ICC protocol, refer to the *ST7 Flash Programming Reference Manual* and to the *ST7 ICC Protocol Reference Manual*.

4.8 REGISTER DESCRIPTION

FLASH CONTROL/STATUS REGISTER (FCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	0	0

This register is reserved for use by Programming Tool software. It controls the Flash programming and erasing operations.

5 CENTRAL PROCESSING UNIT

5.1 INTRODUCTION

This CPU has a full 8-bit architecture and contains six internal registers allowing efficient 8-bit data manipulation.

5.2 MAIN FEATURES

- Enable executing 63 basic instructions
- Fast 8-bit by 8-bit multiply
- 17 main addressing modes (with indirect addressing mode)
- Two 8-bit index registers
- 16-bit stack pointer
- Low power HALT and WAIT modes
- Priority maskable hardware interrupts
- Non-maskable software/hardware interrupts

5.3 CPU REGISTERS

The 6 CPU registers shown in Figure 10 are not present in the memory mapping and are accessed by specific instructions.

Accumulator (A)

The Accumulator is an 8-bit general purpose register used to hold operands and the results of the arithmetic and logic calculations and to manipulate data.

Index Registers (X and Y)

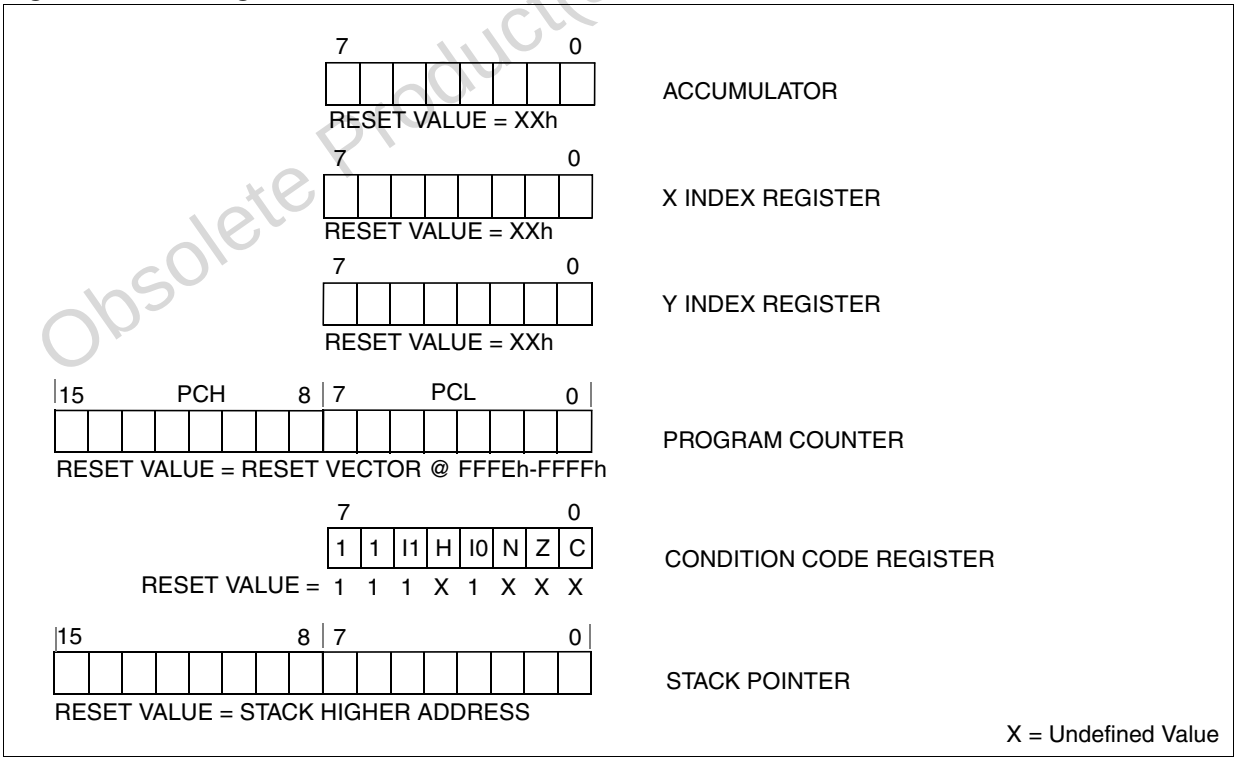
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

The Y register is not affected by the interrupt automatic procedures.

Program Counter (PC)

The program counter is a 16-bit register containing the address of the next instruction to be executed by the CPU. It is made of two 8-bit registers PCL (Program Counter Low which is the LSB) and PCH (Program Counter High which is the MSB).

Figure 10. CPU Registers



CENTRAL PROCESSING UNIT (Cont'd)

Condition Code Register (CC)

Read/Write

Reset Value: 111x1xxx

7							0
1	1	I1	H	I0	N	Z	C

The 8-bit Condition Code register contains the interrupt masks and four flags representative of the result of the instruction just executed. This register can also be handled by the PUSH and POP instructions.

These bits can be individually tested and/or controlled by specific instructions.

Arithmetic Management Bits

Bit 4 = **H** *Half carry*.

This bit is set by hardware when a carry occurs between bits 3 and 4 of the ALU during an ADD or ADC instructions. It is reset by hardware during the same instructions.

0: No half carry has occurred.
1: A half carry has occurred.

This bit is tested using the JRH or JRNH instruction. The H bit is useful in BCD arithmetic subroutines.

Bit 2 = **N** *Negative*.

This bit is set and cleared by hardware. It is representative of the result sign of the last arithmetic, logical or data manipulation. It's a copy of the result 7^{th} bit.
0: The result of the last operation is positive or null.
1: The result of the last operation is negative (i.e. the most significant bit is a logic 1).

This bit is accessed by the JRMI and JRPL instructions.

Bit 1 = **Z** *Zero*.

This bit is set and cleared by hardware. This bit indicates that the result of the last arithmetic, logical or data manipulation is zero.
0: The result of the last operation is different from zero.
1: The result of the last operation is zero.

This bit is accessed by the JREQ and JRNE test instructions.

Bit 0 = **C** *Carry/borrow.*

This bit is set and cleared by hardware and software. It indicates an overflow or an underflow has occurred during the last arithmetic operation.
0: No overflow or underflow has occurred.
1: An overflow or underflow has occurred.

This bit is driven by the SCF and RCF instructions and tested by the JRC and JRNC instructions. It is also affected by the "bit test and branch", shift and rotate instructions.

Interrupt Management Bits

Bit 5,3 = **I1, I0** *Interrupt*

The combination of the I1 and I0 bits gives the current interrupt software priority.

Interrupt Software Priority	I1	I0
Level 0 (main)	1	0
Level 1	0	1
Level 2	0	0
Level 3 (= interrupt disable)	1	1

These two bits are set/cleared by hardware when entering in interrupt. The loaded value is given by the corresponding bits in the interrupt software priority registers (IxSPR). They can be also set/cleared by software with the RIM, SIM, IRET, HALT, WFI and PUSH/POP instructions.

See the interrupt management chapter for more details.

CPU REGISTERS (Cont'd)

STACK POINTER (SP)

Read/Write

Reset Value: 017Fh

15							8
0	0	0	0	0	0	0	1

7							0
1	SP6	SP5	SP4	SP3	SP2	SP1	SP0

The Stack Pointer is a 16-bit register which is always pointing to the next free location in the stack. It is then decremented after data has been pushed onto the stack and incremented before data is popped from the stack (see Figure 11).

Since the stack is 128 bytes deep, the 9 most significant bits are forced by hardware. Following an MCU Reset, or after a Reset Stack Pointer instruction (RSP), the Stack Pointer contains its reset value (the SP6 to SP0 bits are set) which is the stack higher address.

The least significant byte of the Stack Pointer (called S) can be directly accessed by a LD instruction.

Note: When the lower limit is exceeded, the Stack Pointer wraps around to the stack upper limit, without indicating the stack overflow. The previously stored information is then overwritten and therefore lost. The stack also wraps in case of an underflow.

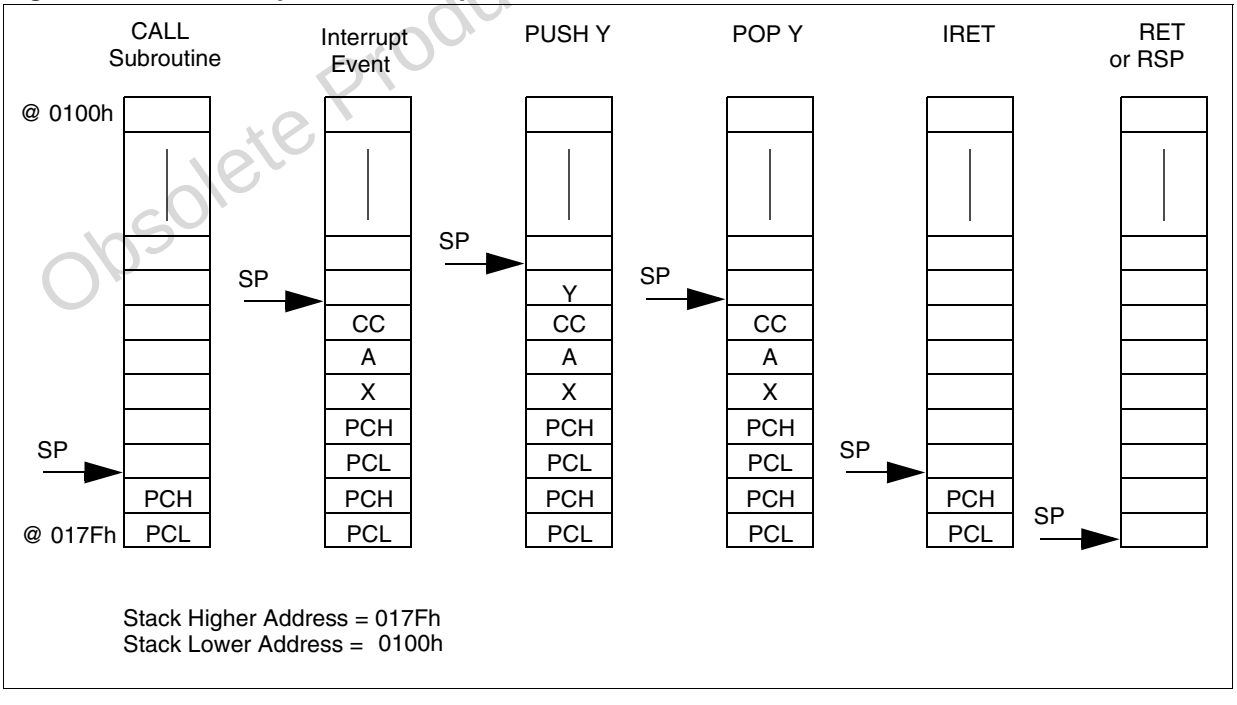
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in Figure 11.

– When an interrupt is received, the SP is decremented and the context is pushed on the stack.

– On return from interrupt, the SP is incremented and the context is popped from the stack.

A subroutine call occupies two locations and an interrupt five locations in the stack area.

Figure 11. Stack Manipulation Example



6 CLOCKS AND RESET

6.1 CLOCK SYSTEM

6.1.1 General Description

The MCU accepts either a Crystal or Ceramic resonator, or an external clock signal to drive the internal oscillator. The internal clock (f_{CPU}) is derived from the external oscillator frequency (f_{OSC}), by dividing by 3 and multiplying by 2. By setting the OSC12/6 bit in the option byte, a 12 MHz external clock can be used giving an internal frequency of 8 MHz while maintaining a 6 MHz clock for USB (refer to Figure 14).

The internal clock signal (f_{CPU}) consists of a square wave with a duty cycle of 50%.

It is further divided by 1, 2, 4 or 8 depending on the Slow Mode Selection bits in the Miscellaneous register (SMS[1:0])

The internal oscillator is designed to operate with an AT-cut parallel resonant quartz or ceramic resonator in the frequency range specified for f_{osc}. The circuit shown in Figure 13 is recommended when using a crystal, and Table 4 lists the recommended capacitors. The crystal and associated components should be mounted as close as possible to the input pins in order to minimize output distortion and start-up stabilization time.

Table 4. Recommended Values for 12 MHz Crystal Resonator

R_{SMAX}	20 Ω	25 Ω	70 Ω
C_{OSCIN}	56pF	47pF	22pF
C_{OSCOUT}	56pF	47pF	22pF
R_P	1-10 MΩ	1-10 MΩ	1-10 MΩ

Note: R_{SMAX} is the equivalent serial resistor of the crystal (see crystal specification).

Note: When a crystal is used, and to not overstress the crystal, ST recommends to add a serial resistor on the OSCOUT pin to limit the drive level in accordance with the crystal manufacturer's specification. Please also refer to Section 12.5.4.

6.1.2 External Clock input

An external clock may be applied to the OSCIN input with the OSCOUT pin not connected, as shown on Figure 12. The t_{OXOV} specifications does not apply when using an external clock input. The equivalent specification of the external clock source should be used instead of t_{OXOV} (see Electrical Characteristics).

6.1.3 Clock Output Pin (MCO)

The internal clock (f_{CPU}) can be output on Port B0 by setting the MCO bit in the Miscellaneous register.

Figure 12. External Clock Source Connections

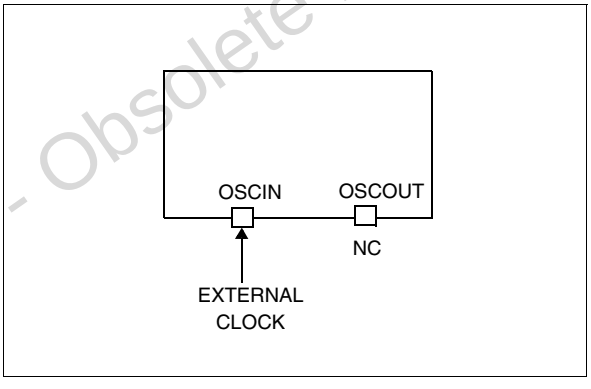


Figure 13. Crystal/Ceramic Resonator

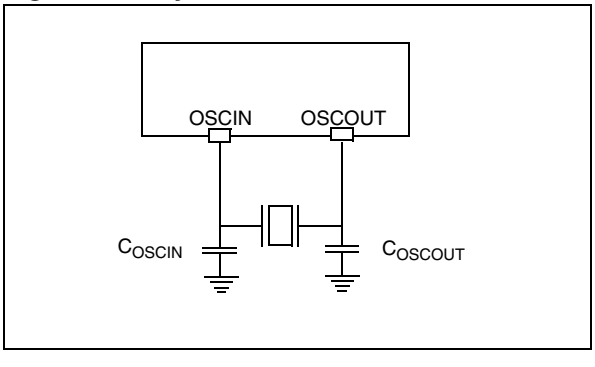
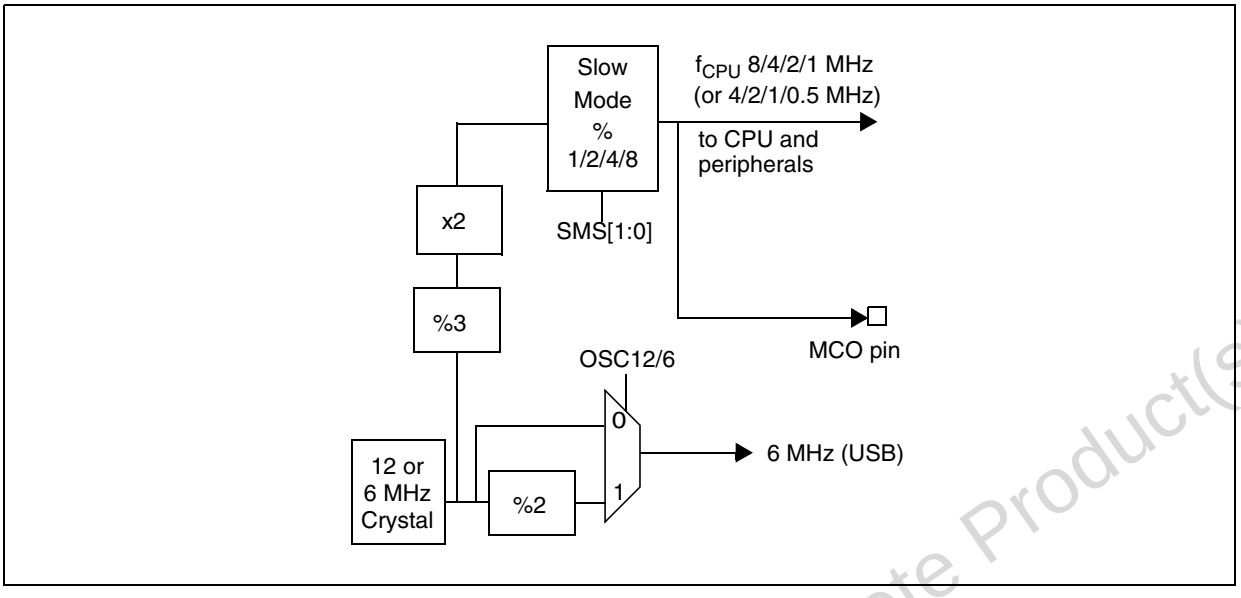


Figure 14. Clock block diagram



6.2 RESET

The Reset procedure is used to provide an orderly software start-up or to exit low power modes.

Three reset modes are provided: a low voltage reset, a watchdog reset and an external reset at the $\overline{\text{RESET}}$ pin.

A reset causes the reset vector to be fetched from addresses FFFEh and FFFFh in order to be loaded into the PC and with program execution starting from this point.

An internal circuitry provides a 514 CPU clock cycle delay from the time that the oscillator becomes active.

Caution: When the ST7 is unprogrammed or fully erased, the Flash is blank and the RESET vector is not programmed. For this reason, it is recommended to keep the RESET pin in low state until programming mode is entered, in order to avoid unwanted behaviour.

6.2.1 Low Voltage Reset

Low voltage reset circuitry generates a reset when V_{DD} is:

- below V_{IT+} when V_{DD} is rising,
- below V_{IT-} when V_{DD} is falling.

During low voltage reset, the $\overline{\text{RESET}}$ pin is held low, thus permitting the MCU to reset other devices.

Notes:

The Low Voltage Detector can be disabled by setting the LVD bit of the Option byte.

It is recommended to make sure that the V_{DD} supply voltage rises monotonously when the device is exiting from Reset, to ensure the application functions properly.

6.2.2 Watchdog Reset

When a watchdog reset occurs, the $\overline{\text{RESET}}$ pin is pulled low permitting the MCU to reset other devices as when low voltage reset (Figure 15).

6.2.3 External Reset

The external reset is an active low input signal applied to the $\overline{\text{RESET}}$ pin of the MCU.
As shown in Figure 18, the $\overline{\text{RESET}}$ signal must stay low for a minimum of one and a half CPU clock cycles.

An internal Schmitt trigger at the $\overline{\text{RESET}}$ pin is provided to improve noise immunity.

Figure 15. Low Voltage Reset functional Diagram

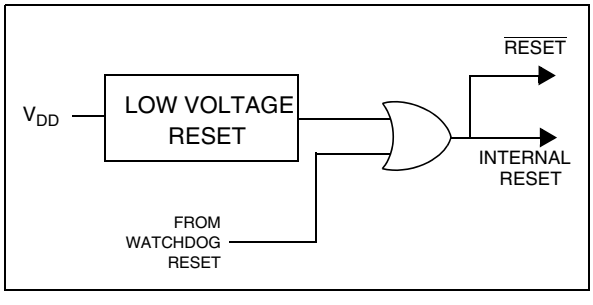
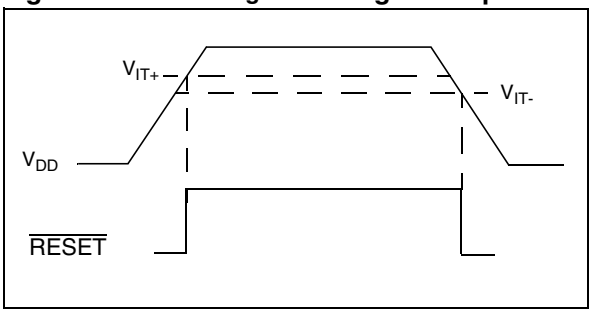


Figure 16. Low Voltage Reset Signal Output



Note: Typical hysteresis (V_{IT+}-V_{IT-}) of 250 mV is expected.

Figure 17. Temporization Timing Diagram after an internal Reset

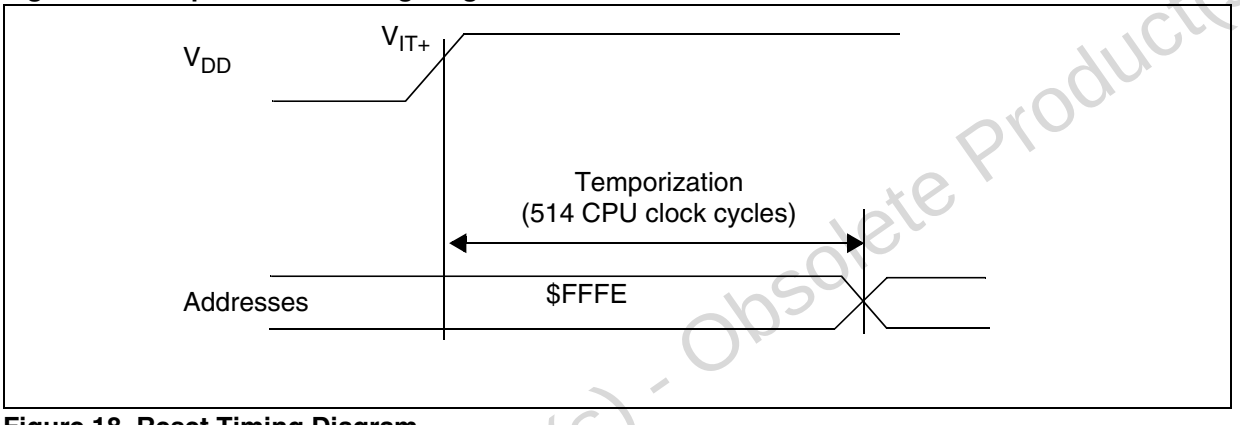
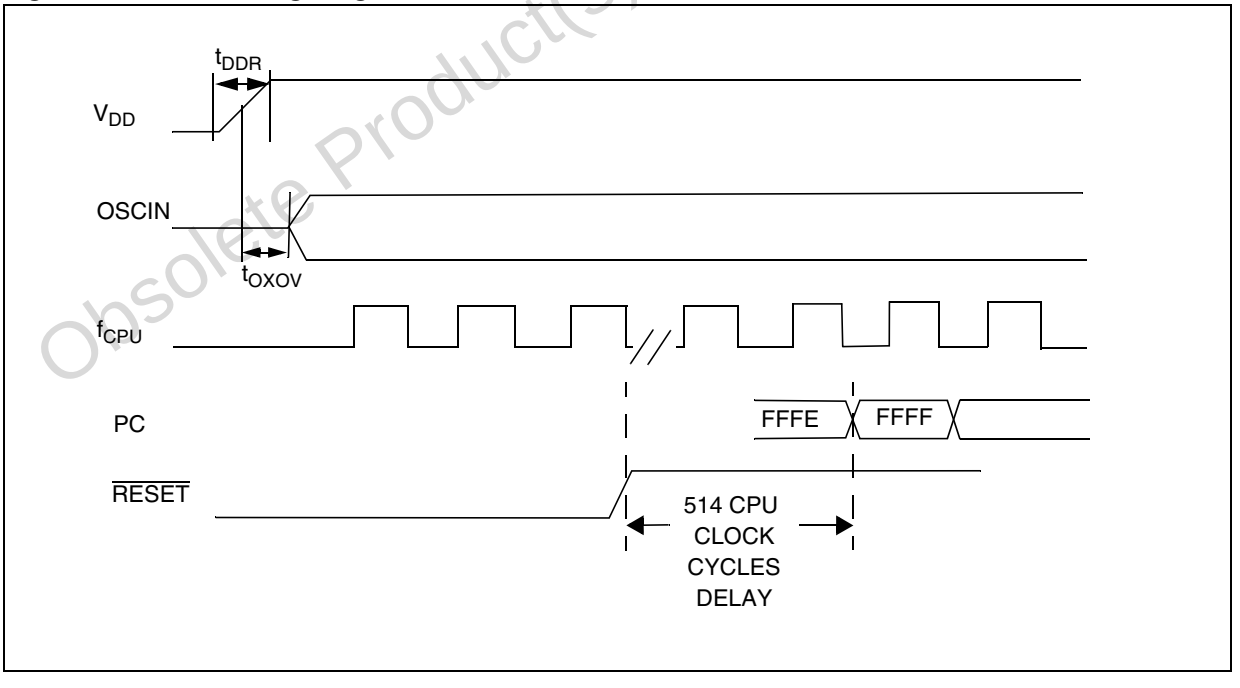
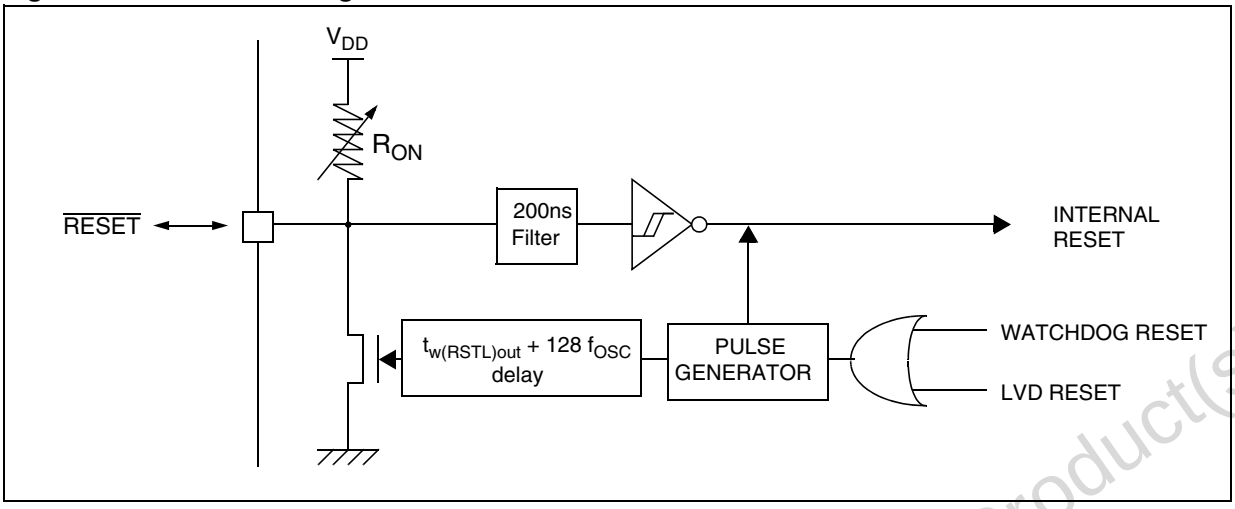


Figure 18. Reset Timing Diagram



Note: Refer to Electrical Characteristics for values of t_{DDR}, t_{OXOV}, V_{IT+} and V_{IT-}.

Figure 19. Reset Block Diagram



Note: The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).

7 INTERRUPTS

7.1 INTRODUCTION

The CPU enhanced interrupt management provides the following features:

- Hardware interrupts
- Software interrupt (TRAP)
- Nested or concurrent interrupt management with flexible interrupt priority and level management:
  - Up to 4 software programmable nesting levels
  - Up to 16 interrupt vectors fixed by hardware
  - 3 non maskable events: RESET, TRAP, TLI

This interrupt management is based on:

– Bit 5 and bit 3 of the CPU CC register (I1:0),

– Interrupt software priority registers (ISPRx),

– Fixed interrupt vector addresses located at the high addresses of the memory map (FFE0h to FFFFh) sorted by hardware priority order.

This enhanced interrupt controller guarantees full upward compatibility with the standard (not nested) CPU interrupt controller.

7.2 MASKING AND PROCESSING FLOW

The interrupt masking is managed by the I1 and I0 bits of the CC register and the ISPRx registers which give the interrupt software priority level of each interrupt vector (see Table 5). The processing flow is shown in Figure 20.

When an interrupt request has to be serviced:

– Normal processing is suspended at the end of the current instruction execution.

– The PC, X, A and CC registers are saved onto the stack.

– I1 and I0 bits of CC register are set according to the corresponding values in the ISPRx registers of the serviced interrupt vector.

– The PC is then loaded with the interrupt vector of the interrupt to service and the first instruction of the interrupt service routine is fetched (refer to “Interrupt Mapping” table for vector addresses).

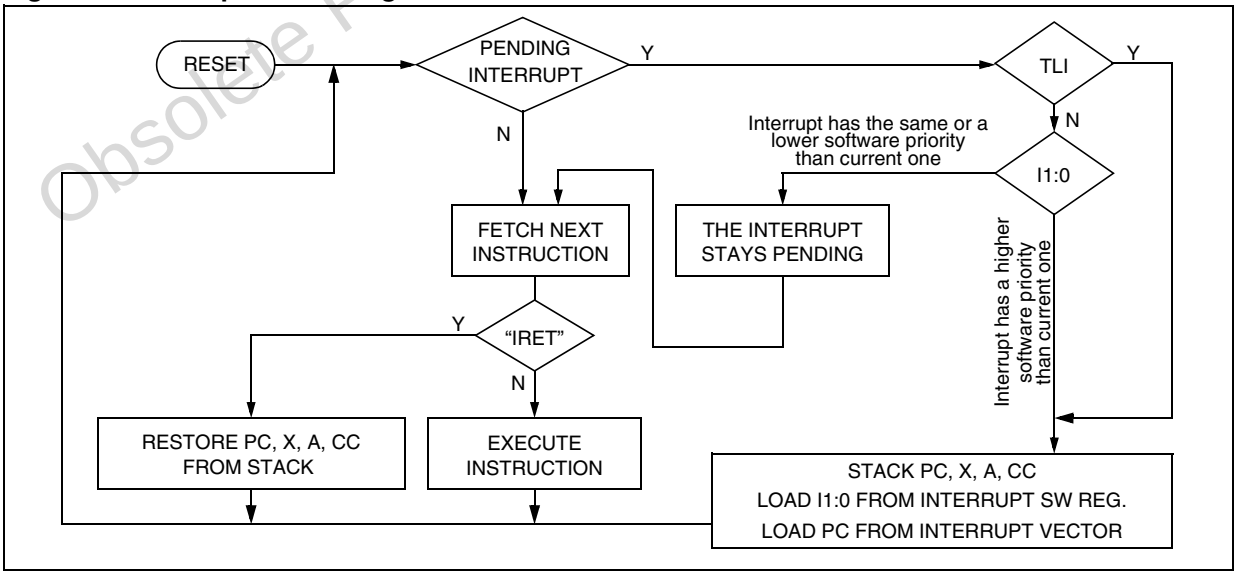
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

Note: As a consequence of the IRET instruction, the I1 and I0 bits will be restored from the stack and the program in the previous level will resume.

Table 5. Interrupt Software Priority Levels

Interrupt software priority	Level	I1	I0
Level 0 (main)	Low	1	0
Level 1	↓	0	1
Level 2	↓	0	0
Level 3 (= interrupt disable)	High	1	1

Figure 20. Interrupt Processing Flowchart



INTERRUPTS (Cont'd)

Servicing Pending Interrupts

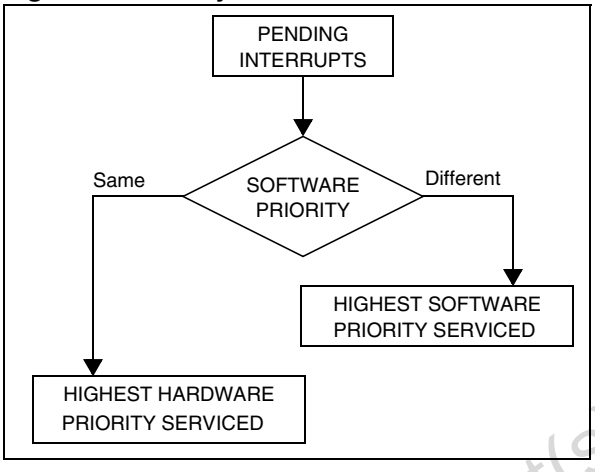
As several interrupts can be pending at the same time, the interrupt to be taken into account is determined by the following two-step process:

– the highest software priority interrupt is serviced,

– if several interrupts have the same software priority then the interrupt with the highest hardware priority is serviced first.

Figure 21 describes this decision process.

Figure 21. Priority Decision Process



When an interrupt request is not serviced immediately, it is latched and then processed when its software priority combined with the hardware priority becomes the highest one.

Note 1: The hardware priority is exclusive while the software one is not. This allows the previous process to succeed with only one interrupt.
Note 2: RESET, TRAP and TLI can be considered as having the highest software priority in the decision process.

Different Interrupt Vector Sources

Two interrupt source types are managed by the CPU interrupt controller: the non-maskable type (RESET, TLI, TRAP) and the maskable type (external or from internal peripherals).

Non-Maskable Sources

These sources are processed regardless of the state of the I1 and I0 bits of the CC register (see Figure 20). After stacking the PC, X, A and CC registers (except for RESET), the corresponding vector is loaded in the PC register and the I1 and I0 bits of the CC are set to disable interrupts (level 3). These sources allow the processor to exit HALT mode.

- TLI (Top Level Hardware Interrupt)

This hardware interrupt occurs when a specific edge is detected on the dedicated TLI pin.
Caution: A TRAP instruction must not be used in a TLI service routine.

- TRAP (Non Maskable Software Interrupt)

This software interrupt is serviced when the TRAP instruction is executed. It will be serviced according to the flowchart in Figure 20 as a TLI.
Caution: TRAP can be interrupted by a TLI.

- RESET

The RESET source has the highest priority in the CPU. This means that the first current routine has the highest software priority (level 3) and the highest hardware priority.
See the RESET chapter for more details.

Maskable Sources

Maskable interrupt vector sources can be serviced if the corresponding interrupt is enabled and if its own interrupt software priority (in ISPRx registers) is higher than the one currently being serviced (I1 and I0 in CC register). If any of these two conditions is false, the interrupt is latched and thus remains pending.

- External Interrupts

External interrupts allow the processor to exit from HALT low power mode.
External interrupt sensitivity is software selectable through the ITRFRE2 register.
External interrupt triggered on edge will be latched and the interrupt request automatically cleared upon entering the interrupt service routine.
If several input pins of a group connected to the same interrupt line are selected simultaneously, these will be logically NANDed.

- Peripheral Interrupts

Usually the peripheral interrupts cause the Device to exit from HALT mode except those mentioned in the "Interrupt Mapping" table.
A peripheral interrupt occurs when a specific flag is set in the peripheral status registers and if the corresponding enable bit is set in the peripheral control register.
The general sequence for clearing an interrupt is based on an access to the status register followed by a read or write to an associated register.
Note: The clearing sequence resets the internal latch. A pending interrupt (i.e. waiting for being serviced) will therefore be lost if the clear sequence is executed.

INTERRUPTS (Cont'd)

7.3 INTERRUPTS AND LOW POWER MODES

All interrupts allow the processor to exit the WAIT low power mode. On the contrary, only external and other specified interrupts allow the processor to exit from the HALT modes (see column "Exit from HALT" in "Interrupt Mapping" table). When several pending interrupts are present while exiting HALT mode, the first one serviced can only be an interrupt with exit from HALT mode capability and it is selected through the same decision process shown in Figure 21.

Note: If an interrupt, that is not able to Exit from HALT mode, is pending with the highest priority when exiting HALT mode, this interrupt is serviced after the first one serviced.

7.4 CONCURRENT & NESTED MANAGEMENT

The following Figure 22 and Figure 23 show two different interrupt management modes. The first is called concurrent mode and does not allow an interrupt to be interrupted, unlike the nested mode in Figure 23. The interrupt hardware priority is given in this order from the lowest to the highest: MAIN, IT4, IT3, IT2, IT1, IT0, TLI. The software priority is given for each interrupt.

Warning: A stack overflow may occur without notifying the software of the failure.

Figure 22. Concurrent Interrupt Management

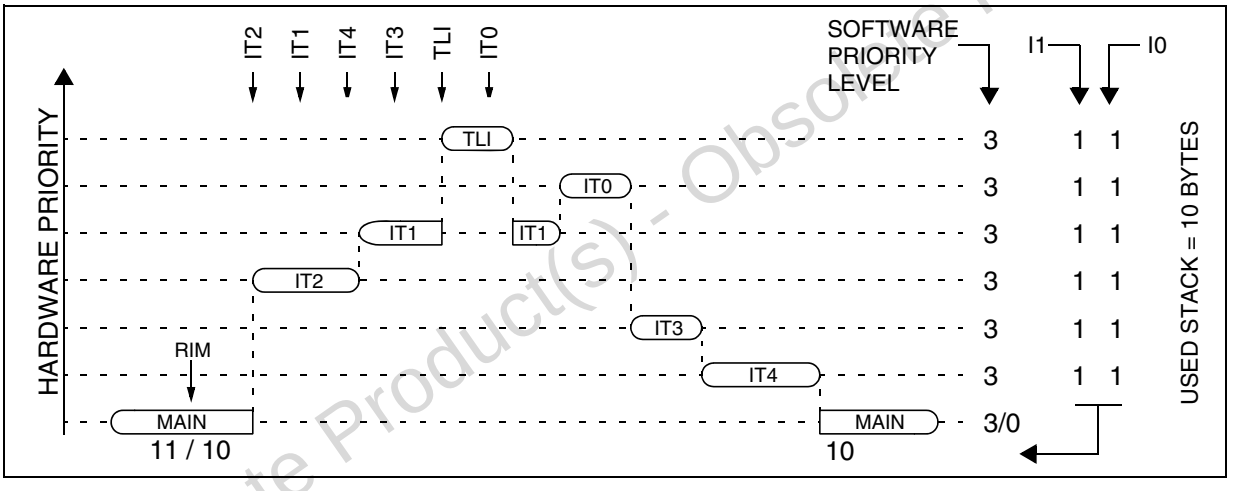
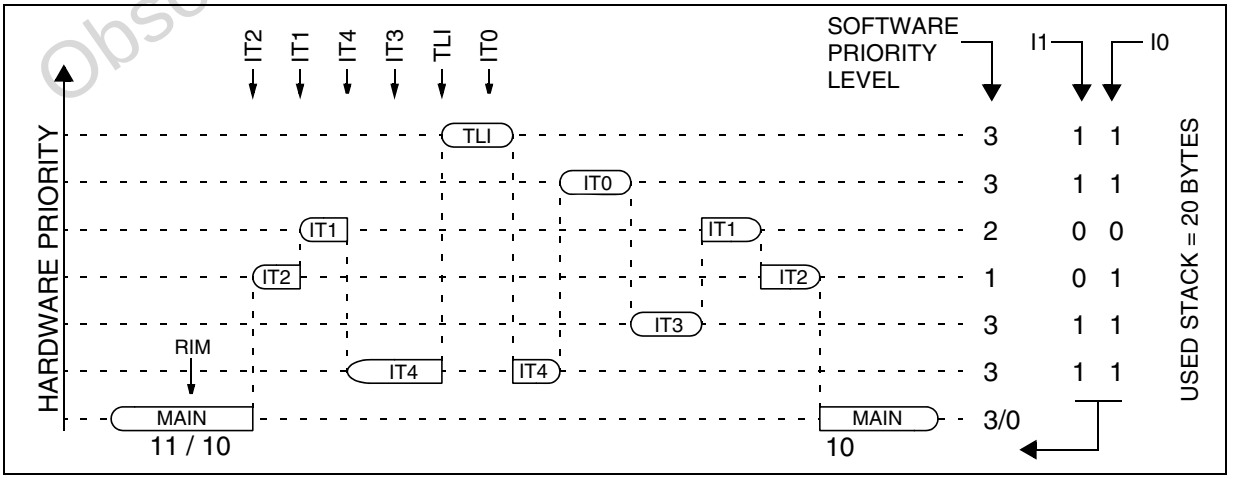


Figure 23. Nested Interrupt Management



INTERRUPTS (Cont'd)

7.5 INTERRUPT REGISTER DESCRIPTION

CPU CC REGISTER INTERRUPT BITS

Read/Write

Reset Value: 111x 1010 (xAh)

7							0
1	1	**I1**	H	**I0**	N	Z	C

Bit 5, 3 = **I1, I0** *Software Interrupt Priority*

These two bits indicate the current interrupt software priority.

Interrupt Software Priority	Level	I1	I0
Level 0 (main)	Low	1	0
Level 1	↓	0	1
Level 2	↓	0	0
Level 3 (= interrupt disable*)	High	1	1

These two bits are set/cleared by hardware when entering in interrupt. The loaded value is given by the corresponding bits in the interrupt software priority registers (ISPRx).

They can be also set/cleared by software with the RIM, SIM, HALT, WFI, IRET and PUSH/POP instructions (see "Interrupt Dedicated Instruction Set" table).

***Note**: TLI, TRAP and RESET events can interrupt a level 3 program.

INTERRUPT SOFTWARE PRIORITY REGISTERS (ISPRX)

Read/Write (bit 7:4 of **ISPR3** are read only)

Reset Value: 1111 1111 (FFh)

7							0
ISPR0	I1_3	I0_3	I1_2	I0_2	I1_1	I0_1	I1_0	I0_0
ISPR1	I1_7	I0_7	I1_6	I0_6	I1_5	I0_5	I1_4	I0_4
ISPR2	I1_11	I0_11	I1_10	I0_10	I1_9	I0_9	I1_8	I0_8
ISPR3	1	1	1	1	I1_13	I0_13	I1_12	I0_12

These four registers contain the interrupt software priority of each interrupt vector.

– Each interrupt vector (except RESET and TRAP) has corresponding bits in these registers where its own software priority is stored. This correspondence is shown in the following table.

Vector address	ISPRx bits
FFFBh-FFFAh	I1_0 and I0_0 bits*
FFF9h-FFF8h	I1_1 and I0_1 bits
...	...
FFE1h-FFE0h	I1_13 and I0_13 bits

– Each I1_x and I0_x bit value in the ISPRx registers has the same meaning as the I1 and I0 bits in the CC register.

– Level 0 can not be written (I1_x=1, I0_x=0). In this case, the previously stored value is kept. (example: previous=CFh, write=64h, result=44h)

The RESET, TRAP and TLI vectors have no software priorities. When one is serviced, the I1 and I0 bits of the CC register are both set.

***Note**: Bits in the ISPRx registers which correspond to the TLI can be read and written but they are not significant in the interrupt process management.

Caution: If the I1_x and I0_x bits are modified while the interrupt x is executed the following behaviour has to be considered: If the interrupt x is still pending (new interrupt or flag not cleared) and the new software priority is higher than the previous one, the interrupt x is re-entered. Otherwise, the software priority stays unchanged up to the next interrupt request (after the IRET of the interrupt x).

INTERRUPTS (Cont'd)

INTERRUPT REGISTER 1 (ITRFRE1)

Address: 0008h - Read/Write

Reset Value: 0000 0000 (00h)

7							0
IT8E	IT7E	IT6E	IT5E	IT4E	IT3E	IT2E	IT1E

Bit 7:0 = **ITiE** *Interrupt Enable*
0: I/O pin free for general purpose I/O
1: ITi external interrupt enabled.

Note: The corresponding interrupt is generated when:

– a rising edge occurs on the IT5/IT6 pins

– a falling edge occurs on the IT1, 2, 3, 4, 7 and 8 pins

INTERRUPT REGISTER 2 (ITRFRE2)

Address: 0039h - Read/Write

Reset Value: 0000 0000 (00h)

7							0
CTL3	CTL2	CTL1	CTL0	IT12E	IT11E	IT10E	IT9E

Bit 7:6 = **CTL[3:2]** *IT[12:11] Interrupt Sensitivity*

These bits are set and cleared by software. They are used to configure the edge and level sensitivity of the IT12 and IT11 external interrupt pins (this means that both must have the same sensitivity).

CTL3	CTL2	IT[12:11] Sensitivity
0	0	Falling edge and low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

Bit 5:4 = **CTL[1:0]** *IT[10:9]1nterrupt Sensitivity*

These bits are set and cleared by software. They are used to configure the edge and level sensitivity of the IT10 and IT9 external interrupt pins (this means that both must have the same sensitivity).

CTL1	CTL0	IT[10:9] Sensitivity
0	0	Falling edge and low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

Bit 3:0 = **ITiE** *Interrupt Enable*
0: I/O pin free for general purpose I/O
1: ITi external interrupt enabled.

INTERRUPTS (Cont'd)

Table 6. Interrupt Mapping

N°	Source Block	Description	Register Label	Priority Order	Exit from HALT	Address Vector
		Reset		Highest Priority	Yes	FFFEh-FFFFh
		TRAP software interrupt			No	FFFCh-FFFDh
0	ICP	FLASH Start programming NMI interrupt			Yes	FFFAh-FFFBh
1	USB	USB End Suspend interrupt	USBISTR		Yes	FFF8h-FFF9h
2	I/O Ports	Port A external interrupts IT[4:1]	ITRFRE1		Yes	FFF6h-FFF7h
3		Port B external interrupts IT[8:5]	ITRFRE1		Yes	FFF4h-FFF5h
4		Port C external interrupts IT[12:9]	ITRFRE2		Yes	FFF2h-FFF3h
5	TBU	Timebase Unit interrupt	TBUCSR		No	FFF0h-FFF1h
6	ART	ART/PWM Timer interrupt	ICCSR		Yes	FFEEh-FFEFh
7	SPI	SPI interrupt vector	SPISR		Yes	FFECh-FFEDh
8	SCI	SCI interrupt vector	SCISR		No	FFEAh-FFEBh
9	USB	USB interrupt vector	USBISTR	Lowest Priority	No	FFE8h-FFE9h
10	ADC	A/D End of conversion interrupt	ADCCSR		No	FFE6h-FFE7h
Reserved area						FFE0h-FFE5h

Table 7. Nested Interrupts Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0032h		Ext. Interrupt Port B		Ext. Interrupt Port A		USB END SUSP		Not Used	
	ISPR0 Reset Value	I1_3 1	I0_3 1	I1_2 1	I0_2 1	I1_1 1	I0_1 1	1	1
0033h		SPI		ART		TBU		Ext. Interrupt Port C	
	ISPR1 Reset Value	I1_7 1	I0_7 1	I1_6 1	I0_6 1	I1_5 1	I0_5 1	I1_4 1	I0_4 1
0034h		Not Used		ADC		USB		SCI	
	ISPR2 Reset Value	I1_11 1	I0_11 1	I1_10 1	I0_10 1	I1_9 1	I0_9 1	I1_8 1	I0_8 1
0035h						Not Used		Not Used	
	ISPR3 Reset Value	1	1	1	1	I1_13 1	I0_13 1	I1_12 1	I0_12 1

8 POWER SAVING MODES

8.1 INTRODUCTION

There are three Power Saving modes. Slow Mode is selected by setting the SMS bits in the Miscellaneous register. Wait and Halt modes may be entered using the WFI and HALT instructions.

After a RESET the normal operating mode is selected by default (RUN mode). This mode drives the device (CPU and embedded peripherals) by means of a master clock which is based on the main oscillator frequency divided by 3 and multiplied by 2 (f_{CPU}).

From Run mode, the different power saving modes may be selected by setting the relevant register bits or by calling the specific ST7 software instruction whose action depends on the oscillator status.

8.1.1 Slow Mode

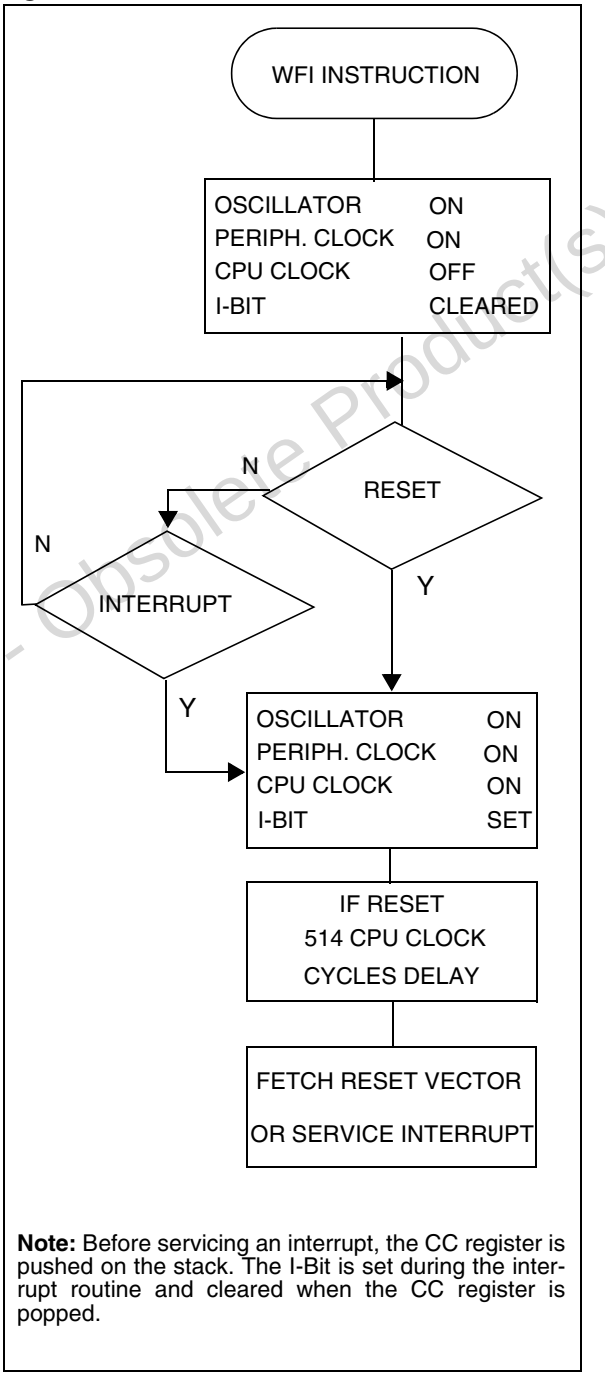
In Slow mode, the oscillator frequency can be divided by a value defined in the Miscellaneous Register. The CPU and peripherals are clocked at this lower frequency. Slow mode is used to reduce power consumption, and enables the user to adapt clock frequency to available supply voltage.

8.2 WAIT MODE

WAIT mode places the MCU in a low power consumption mode by stopping the CPU.
This power saving mode is selected by calling the "WFI" ST7 software instruction.
All peripherals remain active. During WAIT mode, the I bit of the CC register is forced to 0, to enable all interrupts. All other registers and memory remain unchanged. The MCU remains in WAIT mode until an interrupt or Reset occurs, whereupon the Program Counter branches to the starting address of the interrupt or Reset service routine.
The MCU will remain in WAIT mode until a Reset or an Interrupt occurs, causing it to wake up.

Refer to Figure 24.

Figure 24. WAIT Mode Flow Chart



Note: Before servicing an interrupt, the CC register is pushed on the stack. The I-Bit is set during the interrupt routine and cleared when the CC register is popped.

POWER SAVING MODES (Cont'd)

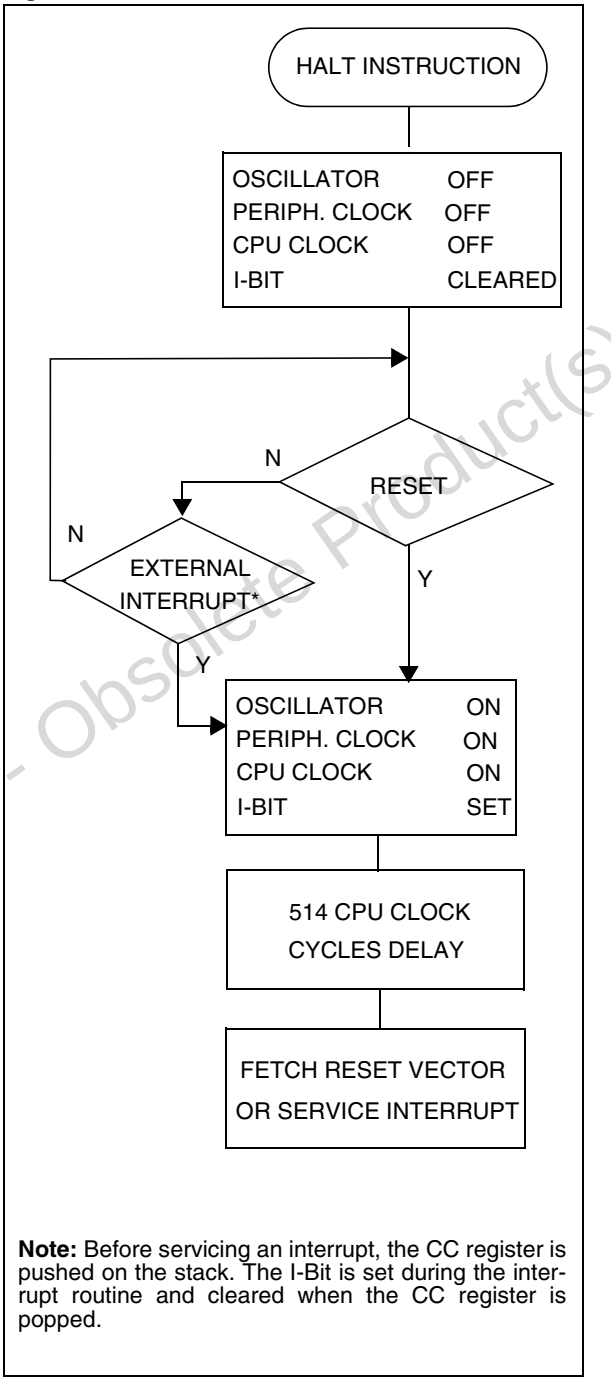
8.3 HALT MODE

The HALT mode is the MCU lowest power consumption mode. The HALT mode is entered by executing the HALT instruction. The internal oscillator is then turned off, causing all internal processing to be stopped, including the operation of the on-chip peripherals.

When entering HALT mode, the I bit in the Condition Code Register is cleared. Thus, any of the external interrupts (ITi or USB end suspend mode), are allowed and if an interrupt occurs, the CPU clock becomes active.

The MCU can exit HALT mode on reception of either an external interrupt on ITi, an end suspend mode interrupt coming from USB peripheral, or a reset. The oscillator is then turned on and a stabilization time is provided before releasing CPU operation. The stabilization time is 514 CPU clock cycles.
After the start up delay, the CPU continues operation by servicing the interrupt which wakes it up or by fetching the reset vector if a reset wakes it up.

Figure 25. HALT Mode Flow Chart



Note: Before servicing an interrupt, the CC register is pushed on the stack. The I-Bit is set during the interrupt routine and cleared when the CC register is popped.

9 I/O PORTS

9.1 INTRODUCTION

The I/O ports offer different functional modes:

transfer of data through digital inputs and outputs and for specific pins:

– Analog signal input (ADC)

– Alternate signal input/output for the on-chip peripherals.

– External interrupt generation

An I/O port is composed of up to 8 pins. Each pin can be programmed independently as digital input or digital output.

9.2 FUNCTIONAL DESCRIPTION

Each port is associated with 2 main registers:

– Data Register (DR)

– Data Direction Register (DDR)

Each I/O pin may be programmed using the corresponding register bits in DDR register: bit x corresponding to pin x of the port. The same correspondence is used for the DR register.

Table 8. I/O Pin Functions

DDR	MODE
0	Input
1	Output

9.2.1 Input Modes

The input configuration is selected by clearing the corresponding DDR register bit.

In this case, reading the DR register returns the digital value applied to the external I/O pin.

Notes:

1. All the inputs are triggered by a Schmitt trigger.
2. When switching from input mode to output mode, the DR register should be written first to output the correct value as soon as the port is configured as an output.

Interrupt function

When an external interrupt function of an I/O pin, is enabled using the ITFRE registers, an event on this I/O can generate an external Interrupt request to the CPU. The interrupt sensitivity is programmable, the options are given in the description of the ITRFRE interrupt registers.

Each pin can independently generate an Interrupt request.

Each external interrupt vector is linked to a dedicated group of I/O port pins (see Interrupts section). If more than one input pin is selected simultaneously as interrupt source, this is logically ANDed and inverted. For this reason, if an event occurs on one of the interrupt pins, it masks the other ones.

9.2.2 Output Mode

The pin is configured in output mode by setting the corresponding DDR register bit (see Table 7).

In this mode, writing “0” or “1” to the DR register applies this digital value to the I/O pin through the latch. Then reading the DR register returns the previously stored value.

Note: In this mode, the interrupt function is disabled.

9.2.3 Alternate Functions

Digital Alternate Functions

When an on-chip peripheral is configured to use a pin, the alternate function is automatically selected. This alternate function takes priority over standard I/O programming. When the signal is coming from an on-chip peripheral, the I/O pin is automatically configured in output mode (push-pull or open drain according to the peripheral).

When the signal is going to an on-chip peripheral, the I/O pin has to be configured in input mode. In this case, the pin state is also digitally readable by addressing the DR register.

Notes:

1. Input pull-up configuration can cause an unexpected value at the alternate peripheral input.
2. When the on-chip peripheral uses a pin as input and output, this pin must be configured as an input (DDR = 0).

***Warning*:** Alternate functions of peripherals must must not be activated when the external interrupts are enabled on the same pin, in order to avoid generating spurious interrupts.

I/O PORTS (Cont'd)

Analog Alternate Functions

When the pin is used as an ADC input, the I/O must be configured as input. The analog multiplexer (controlled by the ADC registers) switches the analog voltage present on the selected pin to the common analog rail which is connected to the ADC input.

It is recommended not to change the voltage level or loading on any port pin while conversion is in progress. Furthermore it is recommended not to have clocking pins located close to a selected analog pin.

Warning: The analog input voltage level must be within the limits stated in the Absolute Maximum Ratings.

9.2.4 I/O Port Implementation

The hardware implementation on each I/O port depends on the settings in the DDR register and specific features of the I/O port such as ADC Input or true open drain.

I/O PORTS (Cont'd)

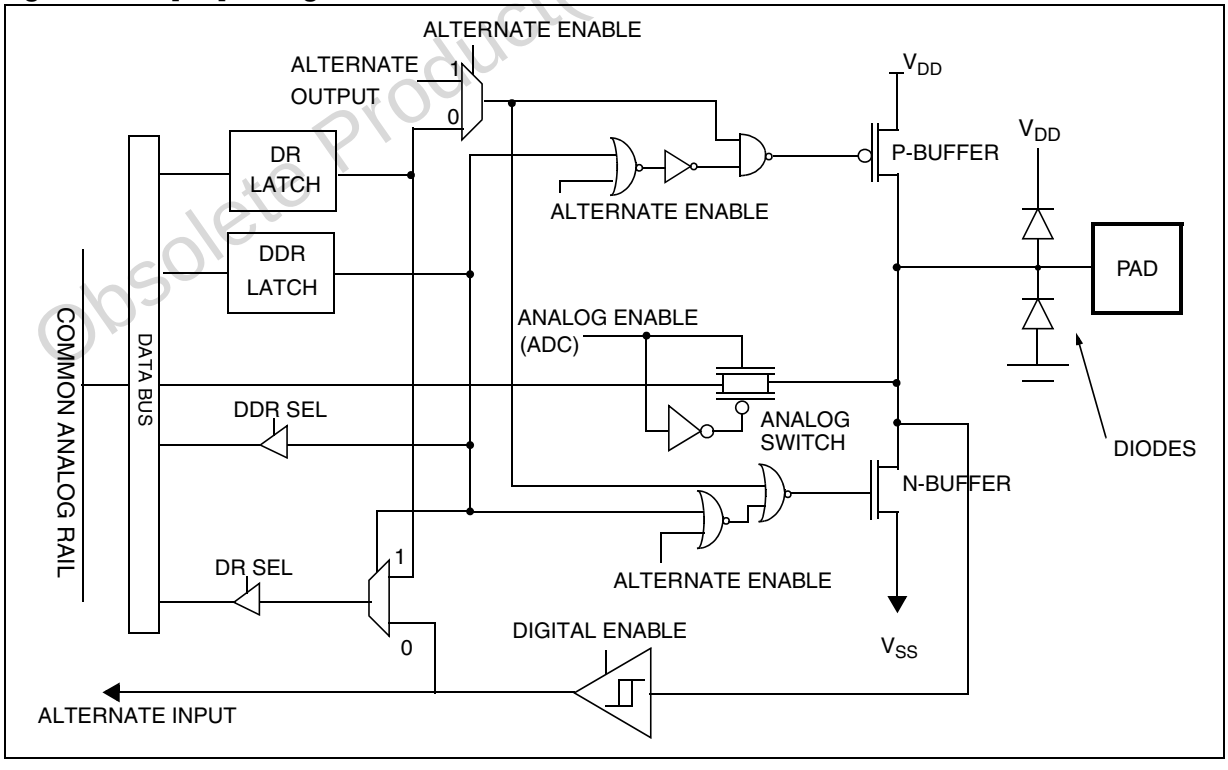
9.2.5 Port A

Table 9. Port A Description

PORT A	I/O		Alternate Function	
	Input*	Output	Signal	Condition
PA0	floating	push-pull	USBOE	USBOE = 1 (MISC)
			IT1 Schmitt triggered input	IT1E = 1 (ITRFRE1)
			AIN0 (ADC)	CS[2:0] = 000 (ADCCSR)
PA1	floating	push-pull	IT2 Schmitt triggered input	IT2E = 1 (ITRFRE1)
			AIN1 (ADC)	CS[2:0] = 001 (ADCCSR)
PA2	floating	push-pull	IT3 Schmitt triggered input	IT3E = 1 (ITRFRE1)
			AIN2 (ADC)	CS[2:0] = 010 (ADCCSR)
PA3	floating	push-pull	IT4 Schmitt triggered input	IT4E = 1 (ITRFRE1)
			AIN3 (ADC)	CS[2:0] = 011 (ADCCSR)
PA4	floating	push-pull	AIN4 (ADC)	CS[2:0] = 100 (ADCCSR)
PA5	floating	push-pull	AIN5 (ADC)	CS[2:0] = 101 (ADCCSR)
PA6	floating	push-pull	AIN6 (ADC)	CS[2:0] = 110 (ADCCSR)
PA7	floating	push-pull	AIN7 (ADC)	CS[2:0] = 111 (ADCCSR)

*Reset State

Figure 26. PA[7:0] Configuration



I/O PORTS (Cont'd)

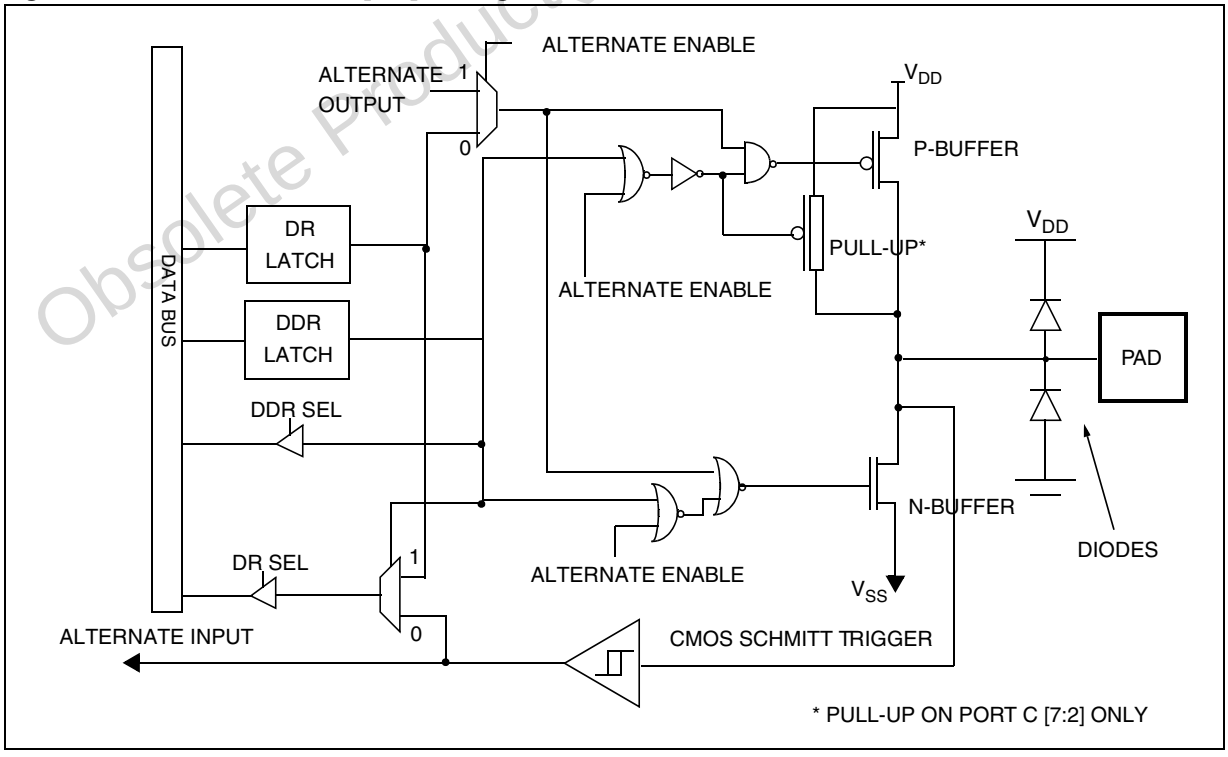
9.2.6 Port B

Table 10. Port B Description

PORT B	I/O		Alternate Function	
	Input*	Output	Signal	Condition
PB0	floating	push-pull (high sink)	MCO (Main Clock Output)	MCO = 1 (MISCR)
PB1	floating	push-pull (high sink)	RDI	SCI enabled
PB2	floating	push-pull (high sink)	TDO	TE = 1 (SCICR2)
PB3	floating	push-pull (high sink)	ARTCLK	EXCL = 1 (ARTCSR)
PB4	floating	push-pull (high sink)	ARTIC1	ART Timer enabled
			IT5 Schmitt triggered input	IT5E = 1 (ITRFRE1)
PB5	floating	push-pull (high sink)	ARTIC2	ART Timer enabled
			IT6 Schmitt triggered input	IT6E = 1 (ITRFRE1)
PB6	floating	push-pull (high sink)	PWM1	OE0 = 1 (PWMCR)
			IT7 Schmitt triggered input	IT7E = 1 (ITRFRE1)
PB7	floating	push-pull (high sink)	PWM2	OE1 = 1 (PWMCR)
			IT8 Schmitt triggered input	IT8E = 1 (ITRFRE1)

*Reset State

Figure 27. Port B and Port C [7:2] Configuration



I/O PORTS (Cont'd)

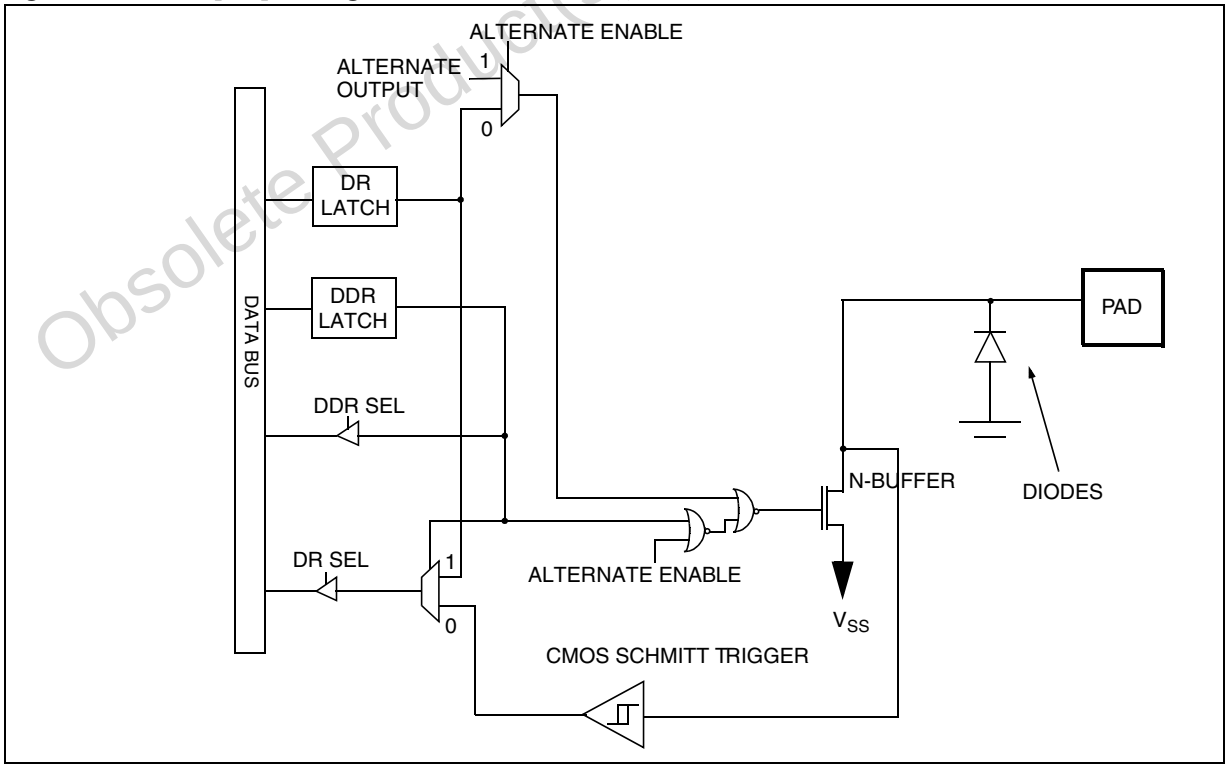
9.2.7 Port C

Table 11. Port C Description

PORT C	I/O		Alternate Function	
	Input*	Output	Signal	Condition
PC0	floating	true open drain		
PC1	floating	true open drain		
PC2	with pull-up	push-pull	IT9 Schmitt triggered input	IT9E = 1 (ITRFRE2)
PC3	with pull-up	push-pull	SCK	SPI enabled
			IT10 Schmitt triggered input	IT10E = 1 (ITRFRE2)
PC4	with pull-up	push-pull	$\overline{SS}$	SPI enabled
			IT11 Schmitt triggered input	IT11E = 1 (ITRFRE2)
PC5	with pull-up	push-pull	MISO	SPI enabled
			IT12 Schmitt triggered input	IT12E = 1 (ITRFRE2)
PC6	with pull-up	push-pull	MOSI	SPI enabled
PC7	with pull-up	push-pull		

*Reset State

Figure 28. Port C[1:0] Configuration



I/O PORTS (Cont'd)

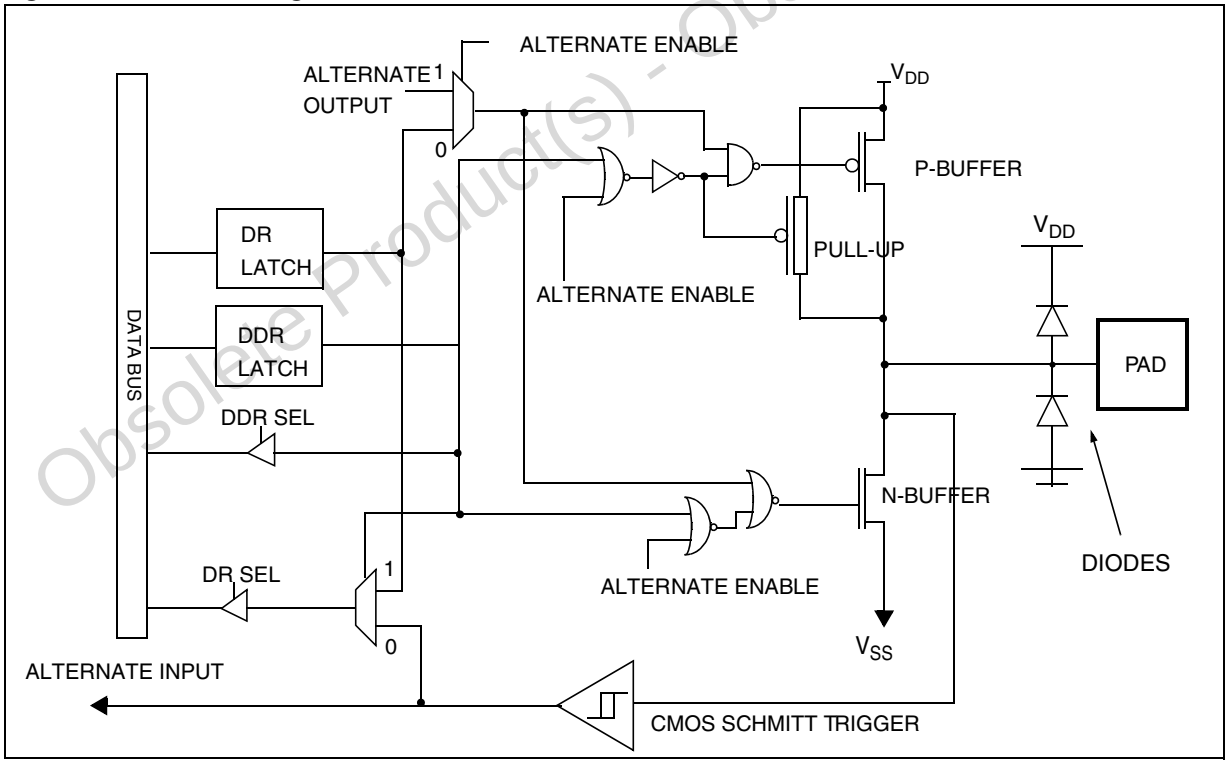
9.2.8 Port D

Table 12. Port D Description

PORT D	I/O		Alternate Function	
	Input*	Output	Signal	Condition
PD0	with pull-up	push-pull		
PD1	with pull-up	push-pull		
PD2	with pull-up	push-pull		
PD3	with pull-up	push-pull		
PD4	with pull-up	push-pull		
PD5	with pull-up	push-pull		
PD6	with pull-up	push-pull		

*Reset State

Figure 29. Port D Configuration



I/O PORTS (Cont'd)

9.2.9 Register Description

DATA REGISTER (DR)

Port x Data Register
PxDR with x = A, B, C or D.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bits 7:0 = **D[7:0]** *Data register 8 bits.*

The DR register has a specific behaviour according to the selected input/output configuration. Writing the DR register is always taken into account even if the pin is configured as an input; this allows to always have the expected level on the pin when toggling to output mode. Reading the DR register returns either the DR register latch content (pin configured as output) or the digital value applied to the I/O pin (pin configured as input).

DATA DIRECTION REGISTER (DDR)

Port x Data Direction Register
PxDDR with x = A, B, C or D.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0

Bits 7:0 = **DD[7:0]** *Data direction register 8 bits.*

The DDR register gives the input/output direction configuration of the pins. Each bit is set and cleared by software.

0: Input mode
1: Output mode

I/O PORTS (Cont'd)

Table 13. I/O Port Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
Reset Value of all I/O port registers		0	0	0	0	0	0	0	0
0000h	**PADR**	MSB							LSB
0001h	**PADDR**								
0002h	**PBDR**	MSB							LSB
0003h	**PBDDR**								
0004h	**PCDR**	MSB							LSB
0005h	**PCDDR**								
0006h	**PDDR**	MSB							LSB
0007h	**PDDDR**								

9.3 MISCELLANEOUS REGISTER

MISCELLANEOUS REGISTER
Read Write
Reset Value - 0000 0000 (00h)

7							0
-	-	-	-	SMS1	SMS0	US-BOE	MCO

Bits 7:4 = Reserved

Bits 3:2 = **SMS[1:0]** *Slow Mode Selection*
These bits select the Slow Mode frequency (depending on the oscillator frequency configured by option byte).

OSC12/6	SMS1	SMS0	Slow Mode Frequency (MHz.)
f_{OSC}= 6 MHz.	0	0	4
	0	1	2
	1	0	1
	1	1	0.5
f_{OSC}= 12 MHz.	0	0	8
	0	1	4
	1	0	2
	1	1	1

Bit 1 = **USBOE** *USB Output Enable*
0: PA0 port free for general purpose I/O
1: USBOE alternate function enabled. The USB output enable signal is output on the PA0 port (at “1” when the ST7 USB is transmitting data).

Bit 0 = **MCO** *Main Clock Out*
0: PB0 port free for general purpose I/O
1: MCO alternate function enabled (f_{CPU} output on PB0 I/O port)

10 ON-CHIP PERIPHERALS

10.1 WATCHDOG TIMER (WDG)

10.1.1 Introduction

The Watchdog timer is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The Watchdog circuit generates an MCU reset on expiry of a programmed time period, unless the program refreshes the counter's contents before the T6 bit becomes cleared.

10.1.2 Main Features

- Programmable free-running downcounter (64 increments of 65536 CPU cycles)
- Programmable reset
- Reset (if watchdog activated) when the T6 bit reaches zero
- Hardware Watchdog selectable by option byte

10.1.3 Functional Description

The counter value stored in the CR register (bits T[6:0]), is decremented every 65,536 machine cycles, and the length of the timeout period can be programmed by the user in 64 increments.

If the watchdog is activated (the WDGA bit is set) and when the 7-bit timer (bits T[6:0]) rolls over from 40h to 3Fh (T6 becomes cleared), it initiates a reset cycle by driving low the reset pin for 30µs.

The application program must write in the CR register at regular intervals during normal operation to prevent an MCU reset. This downcounter is free-running: it counts down even if the watchdog is disabled The value to be stored in the CR register must be between FFh and C0h (see Table 14):

– The WDGA bit is set (watchdog enabled)

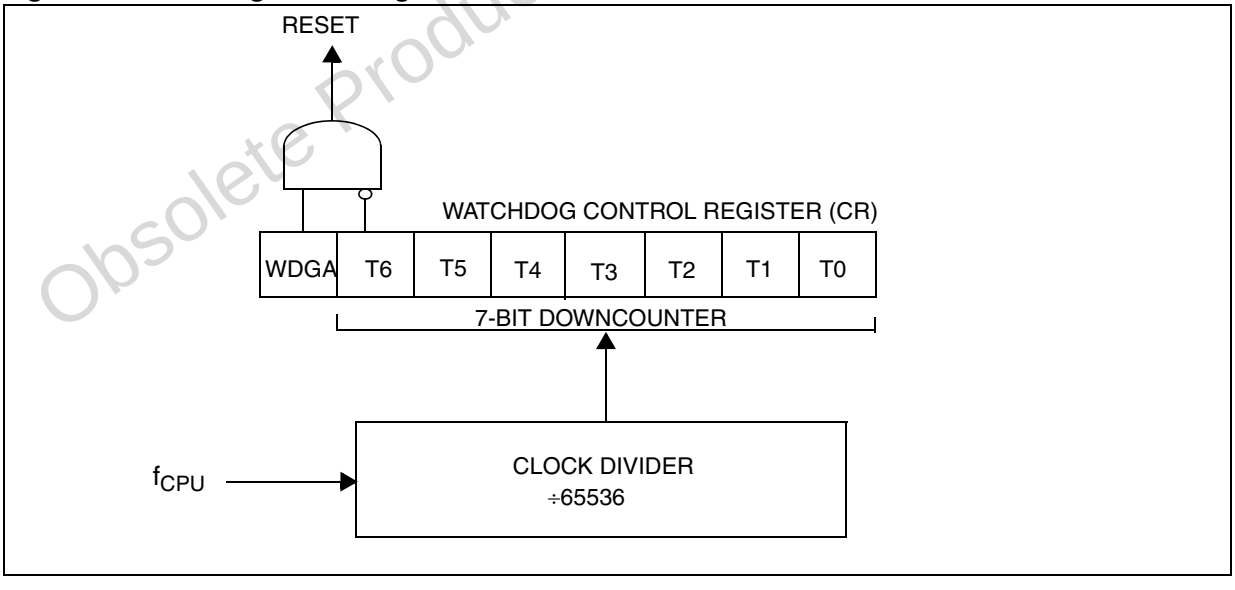
– The T6 bit is set to prevent generating an immediate reset

– The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset.

Table 14.Watchdog Timing (f_{CPU} = 8 MHz)

	CR Register initial value	WDG timeout period (ms)
Max	FFh	524.288
Min	C0h	8.192

Figure 30. Watchdog Block Diagram



WATCHDOG TIMER (Cont'd)

10.1.4 Software Watchdog Option

If Software Watchdog is selected by option byte, the watchdog is disabled following a reset. Once activated it cannot be disabled, except by a reset.

The T6 bit can be used to generate a software reset (the WDGA bit is set and the T6 bit is cleared).

10.1.5 Hardware Watchdog Option

If Hardware Watchdog is selected by option byte, the watchdog is always active and the WDGA bit in the CR is not used.

10.1.6 Low Power Modes

WAIT Instruction

No effect on Watchdog.

HALT Instruction

Halt mode can be used when the watchdog is enabled. When the oscillator is stopped, the WDG stops counting and is no longer able to generate a reset until the microcontroller receives an external interrupt or a reset.

If an external interrupt is received, the WDG restarts counting after 514 CPU clocks. In the case of the Software Watchdog option, if a reset is generated, the WDG is disabled (reset state).

Recommendations

– Make sure that an external event is available to wake up the microcontroller from Halt mode.

– Before executing the HALT instruction, refresh the WDG counter, to avoid an unexpected WDG reset immediately after waking up the microcontroller.

– When using an external interrupt to wake up the microcontroller, reinitialize the corresponding I/O as Input before executing the HALT instruction. The main reason for this is that the I/O may be wrongly configured due to external interference or by an unforeseen logical condition.

– The opcode for the HALT instruction is 0x8E. To avoid an unexpected HALT instruction due to a program counter failure, it is advised to clear all occurrences of the data value 0x8E from memory. For example, avoid defining a constant in ROM with the value 0x8E.

– As the HALT instruction clears the I bit in the CC register to allow interrupts, the user may choose to clear all pending interrupt bits before executing the HALT instruction. This avoids entering other peripheral interrupt routines after executing the external interrupt routine corresponding to the wake-up event (reset or external interrupt).

10.1.7 Interrupts

None.

10.1.8 Register Desc4ription

CONTROL REGISTER (CR)

Read/Write

Reset Value: 0111 1111 (7Fh)

7							0
WDGA	T6	T5	T4	T3	T2	T1	T0

Bit 7 = **WDGA** *Activation bit*.
This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset.
0: Watchdog disabled
1: Watchdog enabled

Note: This bit is not used if the hardware watchdog option is enabled by option byte.

Bits 6:0 = **T[6:0]** *7-bit timer (MSB to LSB).*
These bits contain the decremented value. A reset is produced when it rolls over from 40h to 3Fh (T6 becomes cleared).

Table 15. Watchdog Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0Dh	**WDGCR**	WDGA	T6	T5	T4	T3	T2	T1	T0
	Reset Value	0	1	1	1	1	1	1	1

10.2 PWM AUTO-RELOAD TIMER (ART)

10.2.1 Introduction

The Pulse Width Modulated Auto-Reload Timer on-chip peripheral consists of an 8-bit auto reload counter with compare/capture capabilities and of a 7-bit prescaler clock source.

These resources allow five possible operating modes:

– Generation of up to 2 independent PWM signals

– Output compare and Time base interrupt

– Up to two input capture functions

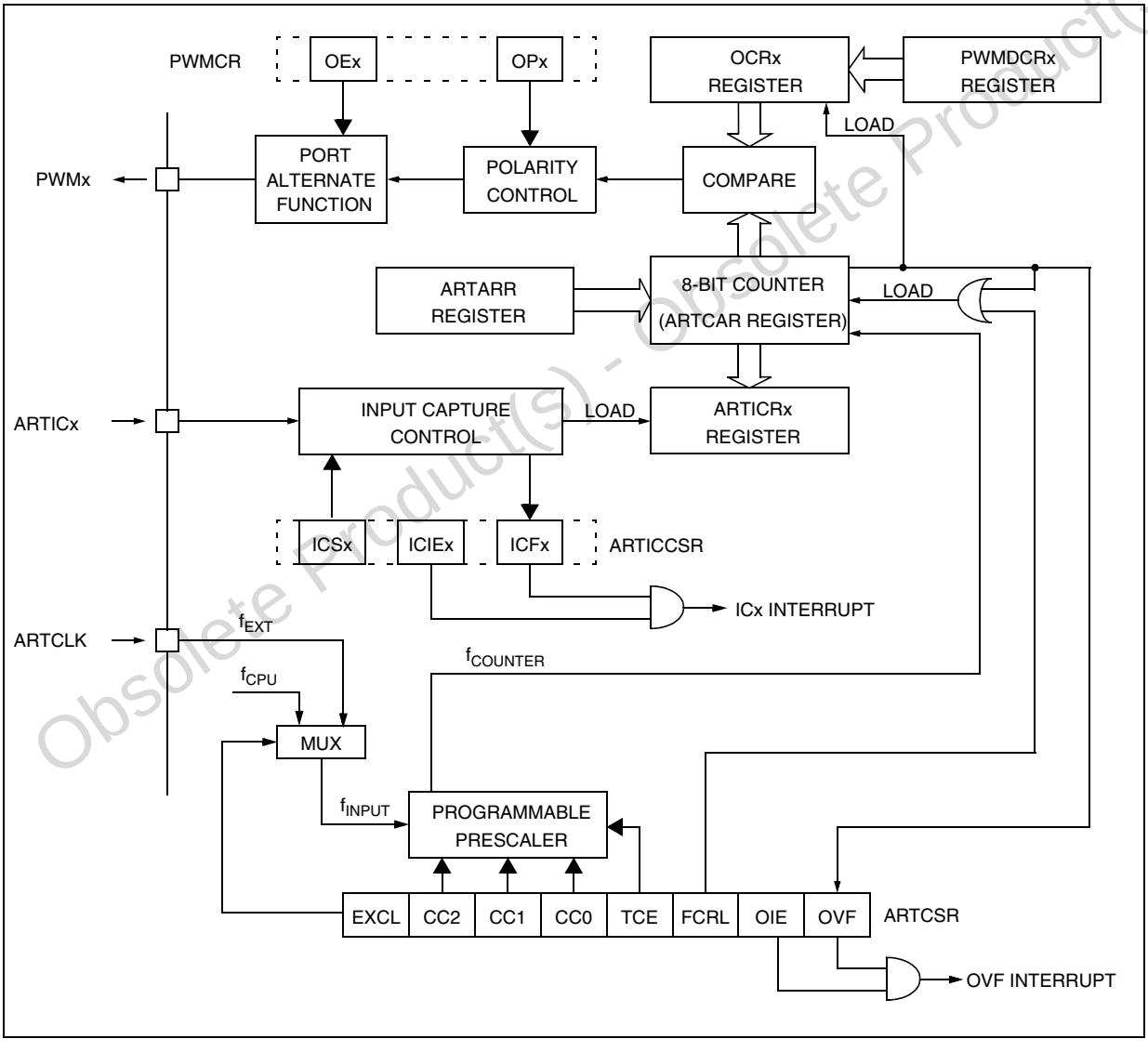
– External event detector

– Up to two external interrupt sources

The three first modes can be used together with a single counter frequency.

The timer can be used to wake up the MCU from WAIT and HALT modes.

Figure 31. PWM Auto-Reload Timer Block Diagram



PWM AUTO-RELOAD TIMER (Cont'd)

10.2.2 Functional Description

Counter

The free running 8-bit counter is fed by the output of the prescaler, and is incremented on every rising edge of the clock signal.

It is possible to read or write the contents of the counter on the fly by reading or writing the Counter Access register (ARTCAR).

When a counter overflow occurs, the counter is automatically reloaded with the contents of the ARTARR register (the prescaler is not affected).

Counter clock and prescaler

The counter clock frequency is given by:

$$f_{COUNTER} = f_{INPUT} / 2^{CC[2:0]}$$

The timer counter's input clock (f_{INPUT}) feeds the 7-bit programmable prescaler, which selects one of the 8 available taps of the prescaler, as defined by CC[2:0] bits in the Control/Status Register (ARTCSR). Thus the division factor of the prescaler can be set to 2^n (where n = 0, 1,..7).

This f_{INPUT} frequency source is selected through the EXCL bit of the ARTCSR register and can be either the f_{CPU} or an external input frequency f_{EXT}.

The clock input to the counter is enabled by the TCE (Timer Counter Enable) bit in the ARTCSR register. When TCE is reset, the counter is stopped and the prescaler and counter contents are frozen. When TCE is set, the counter runs at the rate of the selected clock source.

Counter and Prescaler Initialization

After RESET, the counter and the prescaler are cleared and $f_{INPUT} = f_{CPU}$.

The counter can be initialized by:

– Writing to the ARTARR register and then setting the FCRL (Force Counter Re-Load) and the TCE (Timer Counter Enable) bits in the ARTCSR register.

– Writing to the ARTCAR counter access register,

In both cases the 7-bit prescaler is also cleared, whereupon counting will start from a known value.

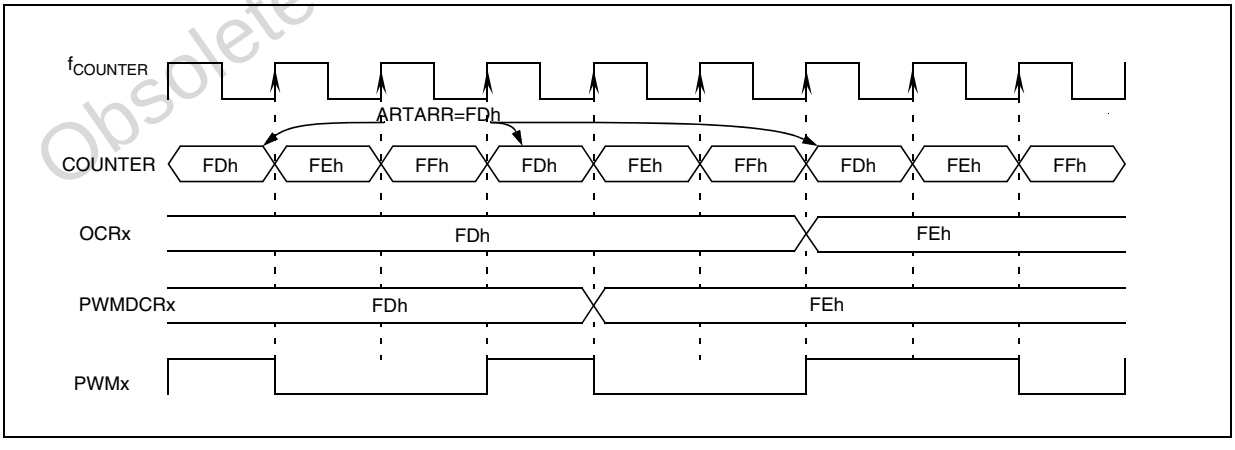
Direct access to the prescaler is not possible.

Output compare control

The timer compare function is based on four different comparisons with the counter (one for each PWMx output). Each comparison is made between the counter value and an output compare register (OCRx) value. This OCRx register can not be accessed directly, it is loaded from the duty cycle register (PWMDCRx) at each overflow of the counter.

This double buffering method avoids glitch generation when changing the duty cycle on the fly.

Figure 32. Output compare control



PWM AUTO-RELOAD TIMER (Cont'd)

Independent PWM signal generation

This mode allows up to two Pulse Width Modulated signals to be generated on the PWMx output pins with minimum core processing overhead. This function is stopped during HALT mode.

Each PWMx output signal can be selected independently using the corresponding OEx bit in the PWM Control register (PWMCR). When this bit is set, the corresponding I/O pin is configured as output push-pull alternate function.

The PWM signals all have the same frequency which is controlled by the counter period and the ARTARR register value.

$$f_{PWM} = f_{COUNTER} / (256 - ARTARR)$$

When a counter overflow occurs, the PWMx pin level is changed depending on the corresponding OPx (output polarity) bit in the PWMCR register. When the counter reaches the value contained in one of the output compare register (OCRx) the corresponding PWMx pin level is restored.

It should be noted that the reload values will also affect the value and the resolution of the duty cycle of the PWM output signal. To obtain a signal on a PWMx pin, the contents of the OCRx register must be greater than the contents of the ARTARR register.

The maximum available resolution for the PWMx duty cycle is:

$$Resolution = 1 / (256 - ARTARR)$$

Note: To get the maximum resolution (1/256), the ARTARR register must be 0. With this maximum resolution, 0% and 100% can be obtained by changing the polarity.

Figure 33. PWM Auto-reload Timer Function

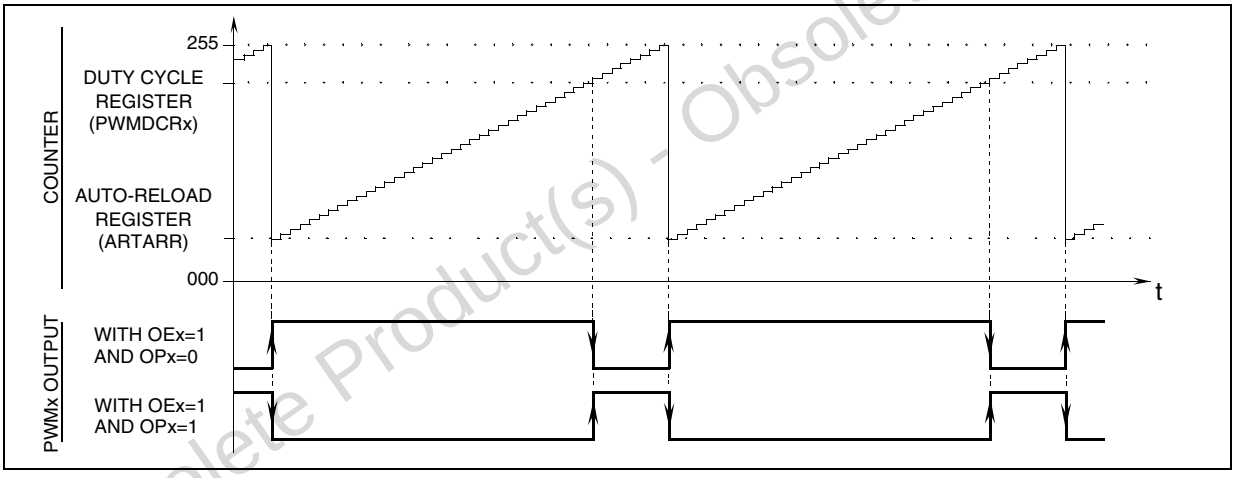
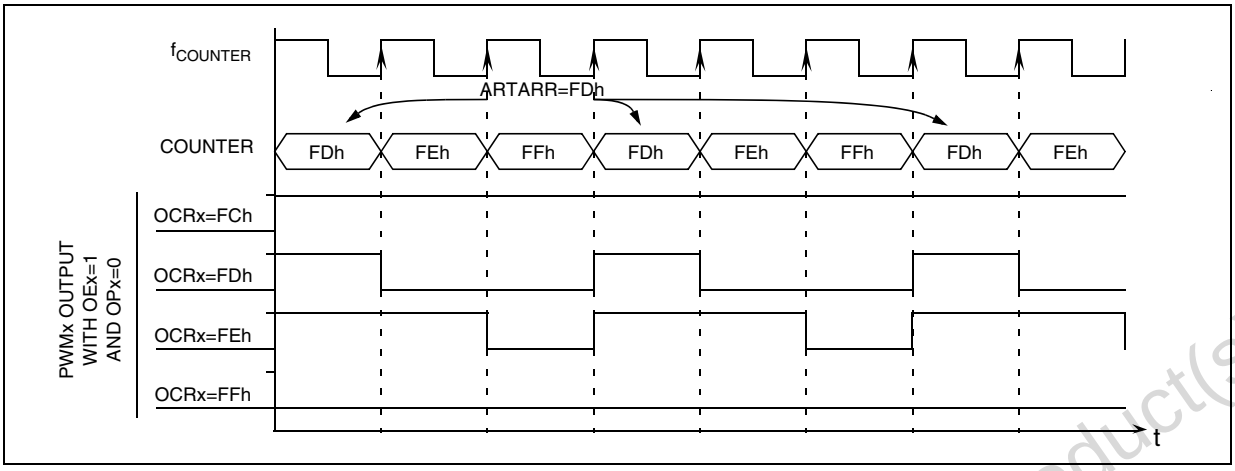


Figure 34. PWM Signal from 0% to 100% Duty Cycle



PWM AUTO-RELOAD TIMER (Cont'd)

Output compare and Time base interrupt

On overflow, the OVF flag of the ARTCSR register is set and an overflow interrupt request is generated if the overflow interrupt enable bit, OIE, in the ARTCSR register, is set. The OVF flag must be reset by the user software. This interrupt can be used as a time base in the application.

External clock and event detector mode

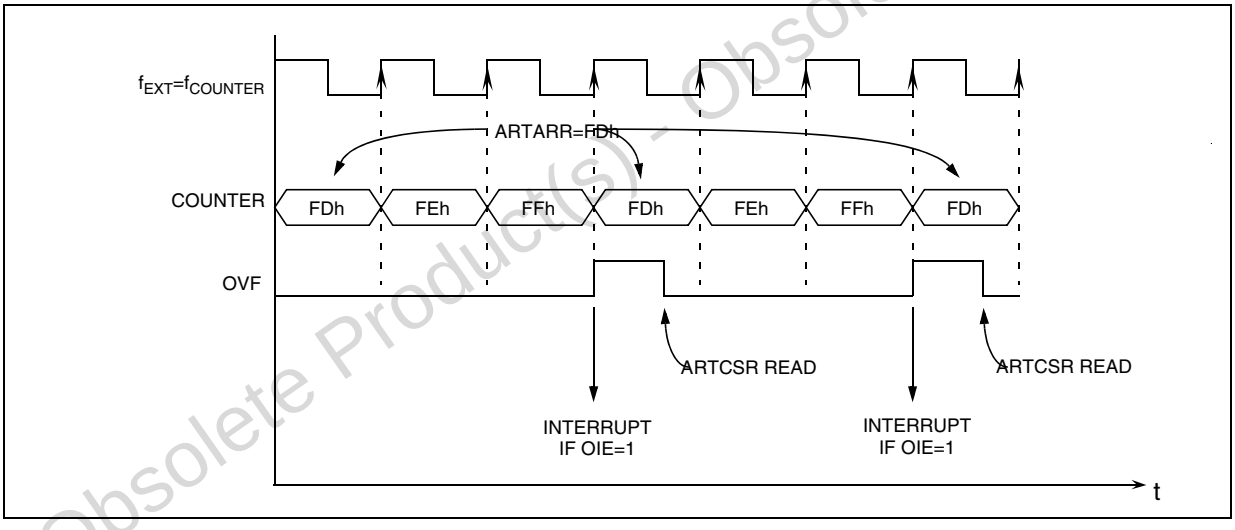
Using the f_{EXT} external prescaler input clock, the auto-reload timer can be used as an external clock event detector. In this mode, the ARTARR register is used to select the n_{EVENT} number of events to be counted before setting the OVF flag.

$$n_{EVENT} = 256 - ARTARR$$

When entering HALT mode while f_{EXT} is selected, all the timer control registers are frozen but the counter continues to increment. If the OIE bit is set, the next overflow of the counter will generate an interrupt which wakes up the MCU.

Caution: If HALT mode is used in the application, prior to executing the HALT instruction, the counter must be disabled by clearing the TCE bit in the ARTCSR register to avoid spurious counter increments.

Figure 35. External Event Detector Example (3 counts)



PWM AUTO-RELOAD TIMER (Cont'd)

Input capture function

This mode allows the measurement of external signal pulse widths through ICRx registers.

Each input capture can generate an interrupt independently on a selected input signal transition. This event is flagged by a set of the corresponding CFx bits of the Input Capture Control/Status register (ICCSR).

These input capture interrupts are enabled through the CIEx bits of the ICCSR register.

The active transition (falling or rising edge) is software programmable through the CSx bits of the ICCSR register.

The read only input capture registers (ICRx) are used to latch the auto-reload counter value when a transition is detected on the ARTICx pin (CFx bit set in ICCSR register). After fetching the interrupt vector, the CFx flags can be read to identify the interrupt source.

Note: After a capture detection, data transfer in the ICRx register is inhibited until the ARTICCSR register is read (clearing the CFx bit).
The timer interrupt remains pending while the CFx flag is set when the interrupt is enabled (CIEx bit set). This means, the ARTICCSR register has to be read at each capture event to clear the CFx flag.

The timing resolution is given by auto-reload counter cycle time ($1/f_{COUNTER}$).

During HALT mode, input capture is inhibited (the ICRx is never re-loaded) and only the external interrupt capability can be used.

External interrupt capability

This mode allows the Input capture capabilities to be used as external interrupt sources.

The edge sensitivity of the external interrupts is programmable (CSx bit of ICCSR register) and they are independently enabled through CIEx bits of the ICCSR register. After fetching the interrupt vector, the CFx flags can be read to identify the interrupt source.

The interrupts are synchronized on the counter clock rising edge (Figure 36).

During HALT mode, the external interrupts can still be used to wake up the micro (if CIEx bit is set).

Figure 36. ART External Interrupt

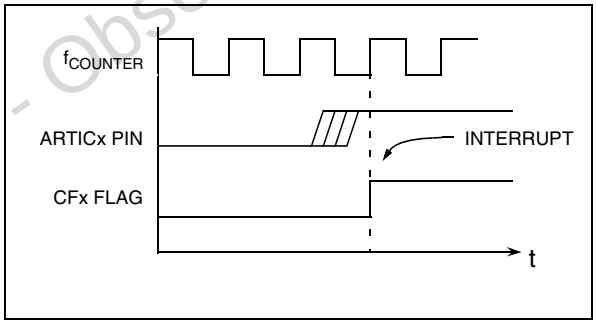
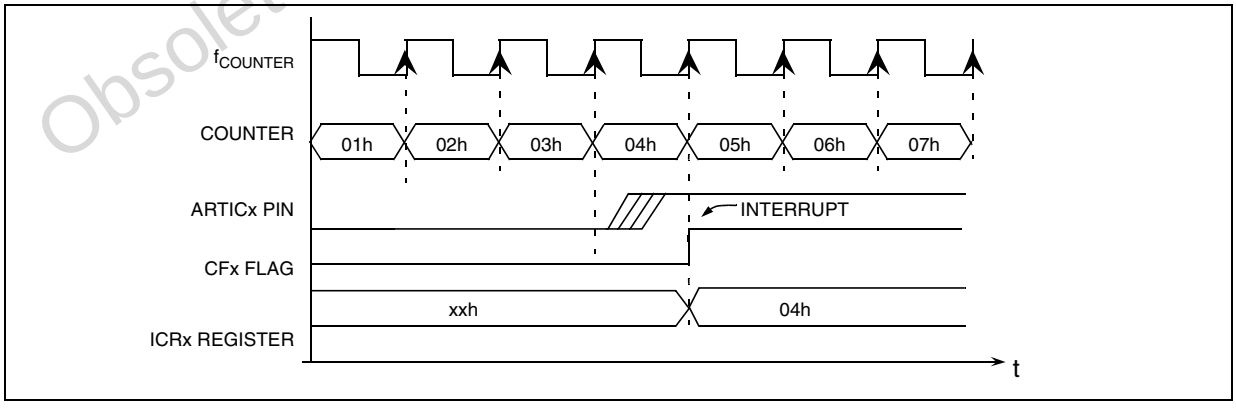


Figure 37. Input Capture Timing Diagram



PWM AUTO-RELOAD TIMER (Cont'd)

10.2.3 Register Description

CONTROL / STATUS REGISTER (CSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
EXCL	CC2	CC1	CC0	TCE	FCRL	OIE	OVF

Bit 7 = **EXCL** *External Clock*
This bit is set and cleared by software. It selects the input clock for the 7-bit prescaler.
0: CPU clock.
1: External clock.

Bit 6:4 = **CC[2:0]** *Counter Clock Control*
These bits are set and cleared by software. They determine the prescaler division ratio from f_{INPUT}.

$f_{COUNTER}$	With f_{INPUT}=8 MHz	CC2	CC1	CC0
f_{INPUT}	8 MHz	0	0	0
f_{INPUT} / 2	4 MHz	0	0	1
f_{INPUT} / 4	2 MHz	0	1	0
f_{INPUT} / 8	1 MHz	0	1	1
f_{INPUT} / 16	500 KHz	1	0	0
f_{INPUT} / 32	250 KHz	1	0	1
f_{INPUT} / 64	125 KHz	1	1	0
f_{INPUT} / 128	62.5 KHz	1	1	1

Bit 3 = **TCE** *Timer Counter Enable*
This bit is set and cleared by software. It puts the timer in the lowest power consumption mode.
0: Counter stopped (prescaler and counter frozen).
1: Counter running.

Bit 2 = **FCRL** *Force Counter Re-Load*
This bit is write-only and any attempt to read it will yield a logical zero. When set, it causes the contents of ARR register to be loaded into the counter, and the content of the prescaler register to be cleared in order to initialize the timer before starting to count.

Bit 1 = **OIE** *Overflow Interrupt Enable*
This bit is set and cleared by software. It allows to enable/disable the interrupt which is generated when the OVF bit is set.
0: Overflow Interrupt disable.
1: Overflow Interrupt enable.

Bit 0 = **OVF** *Overflow Flag*
This bit is set by hardware and cleared by software reading the CSR register. It indicates the transition of the counter from FFh to the ARR value.
0: New transition not yet reached
1: Transition reached

COUNTER ACCESS REGISTER (CAR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0

Bit 7:0 = **CA[7:0]** *Counter Access Data*

These bits can be set and cleared either by hardware or by software. The CAR register is used to read or write the auto-reload counter "on the fly" (while it is counting).

AUTO-RELOAD REGISTER (ARR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0

Bit 7:0 = **AR[7:0]** *Counter Auto-Reload Data*

These bits are set and cleared by software. They are used to hold the auto-reload value which is automatically loaded in the counter when an overflow occurs. At the same time, the PWM output levels are changed according to the corresponding OPx bit in the PWMCR register.

This register has two PWM management functions:

- Adjusting the PWM frequency
- Setting the PWM duty cycle resolution

PWM Frequency vs. Resolution:

ARR value	Resolution	f_{PWM}	
		Min	Max
0	8-bit	~0.244-KHz	31.25-KHz
[0..127]	> 7-bit	~0.244-KHz	62.5-KHz
[128..191]	> 6-bit	~0.488-KHz	125-KHz
[192..223]	> 5-bit	~0.977-KHz	250-KHz
[224..239]	> 4-bit	~1.953-KHz	500-KHz

PWM AUTO-RELOAD TIMER (Cont'd)

PWM CONTROL REGISTER (PWMCR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	OE1	OE0	0	0	OP1	OP0

Bit 7:6 = Reserved.

Bit 5:4 = **OE[1:0]** *PWM Output Enable*
These bits are set and cleared by software. They enable or disable the PWM output channels independently acting on the corresponding I/O pin.
0: PWM output disabled.
1: PWM output enabled.

Bit 3:2 = Reserved.

Bit 1:0 = **OP[1:0]** *PWM Output Polarity*
These bits are set and cleared by software. They independently select the polarity of the two PWM output signals.

PWMx output level		OPx
Counter <= OCRx	Counter > OCRx	
1	0	0
0	1	1

Notes:
– When an OPx bit is modified, the PWMx output signal polarity is immediately reversed.
– If DCRx=FFh then the output level is always 0.
– If DCRx=00h then the output level is always 1.

DUTY CYCLE REGISTERS (DCRx)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0

Bit 7:0 = **DC[7:0]** *Duty Cycle Data*

These bits are set and cleared by software.

A DCRx register is associated with the OCRx register of each PWM channel to determine the second edge location of the PWM signal (the first edge location is common to all channels and given by the ARR register). These DCR registers allow the duty cycle to be set independently for each PWM channel.

PWM AUTO-RELOAD TIMER (Cont'd)

INPUT CAPTURE CONTROL / STATUS REGISTER (ARTICCSR)

Read/Write (except bits 1:0 read and clear)

Reset Value: 0000 0000 (00h)

7							0
0	0	CS2	CS1	CIE2	CIE1	CF2	CF1

Bit 7:6 = Reserved, always read as 0.

Bit 5:4 = **CS[2:1]** *Capture Sensitivity*
These bits are set and cleared by software. They determine the trigger event polarity on the corresponding input capture channel.
0: Falling edge triggers capture on channel x.
1: Rising edge triggers capture on channel x.

Bit 3:2 = **CIE[2:1]** *Capture Interrupt Enable*
These bits are set and cleared by software. They enable or disable the Input capture channel interrupts independently.
0: Input capture channel x interrupt disabled.
1: Input capture channel x interrupt enabled.

Bit 1:0 = **CF[2:1]** *Capture Flag*
These bits are set by hardware when a capture occurs and cleared by hardware when software reads the ARTICCSR register. Each CFx bit indicates that an input capture x has occurred.
0: No input capture on channel x.
1: An input capture has occurred on channel x.

INPUT CAPTURE REGISTERS (ARTICRx)

Read only

Reset Value: 0000 0000 (00h)

7							0
IC7	IC6	IC5	IC4	IC3	IC2	IC1	IC0

Bit 7:0 = **IC[7:0]** *Input Capture Data*

These read only bits are set and cleared by hardware. An ARTICRx register contains the 8-bit auto-reload counter value transferred by the input capture channel x event.

PWM AUTO-RELOAD TIMER (Cont'd)

Table 16. PWM Auto-Reload Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0014h	**PWMDCR1**	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
	Reset Value	0	0	0	0	0	0	0	0
0015h	**PWMDCR0**	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
	Reset Value	0	0	0	0	0	0	0	0
0016h	**PWMCR**	0	0	OE1	OE0	0	0	OP1	OP0
	Reset Value	0	0	0	0	0	0	0	0
0017h	**ARTCSR**	EXCL	CC2	CC1	CC0	TCE	FCRL	OIE	OVF
	Reset Value	0	0	0	0	0	0	0	0
0018h	**ARTCAR**	CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0
	Reset Value	0	0	0	0	0	0	0	0
0019h	**ARTARR**	AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
	Reset Value	0	0	0	0	0	0	0	0
001Ah	**ARTICCSR**			CS2	CS1	CIE2	CIE1	CF2	CF1
	Reset Value	0	0	0	0	0	0	0	0
001Bh	**ARTICR1**	IC7	IC6	IC5	IC4	IC3	IC2	IC1	IC0
	Reset Value	0	0	0	0	0	0	0	0
001Ch	**ARTICR2**	IC7	IC6	IC5	IC4	IC3	IC2	IC1	IC0
	Reset Value	0	0	0	0	0	0	0	0

10.3 TIMEBASE UNIT (TBU)

10.3.1 Introduction

The Timebase unit (TBU) can be used to generate periodic interrupts.

10.3.2 Main Features

- 8-bit upcounter
- Programmable prescaler
- Period between interrupts: max. 8.1ms (at 8 MHz f_{CPU})
- Maskable interrupt
- Cascadable with PWM/ART TImer

10.3.3 Functional Description

The TBU operates as a free-running upcounter.

When the TCEN bit in the TBUCSR register is set by software, counting starts at the current value of the TBUCV register. The TBUCV register is incremented at the clock rate output from the prescaler selected by programming the PR[2:0] bits in the TBUCSR register.

When the counter rolls over from FFh to 00h, the OVF bit is set and an interrupt request is generated if ITE is set.

The user can write a value at any time in the TBUCV register.

If the cascading option is selected (CAS bit=1 in the TBUCSR register), the TBU and the ART TImer counters act together as a 16-bit counter. In this case, the TBUCV register is the high order byte, the ART counter (ARTCAR register) is the low order byte. Counting is clocked by the ART timer clock (Refer to the description of the ART Timer ARTCSR register).

10.3.4 Programming Example

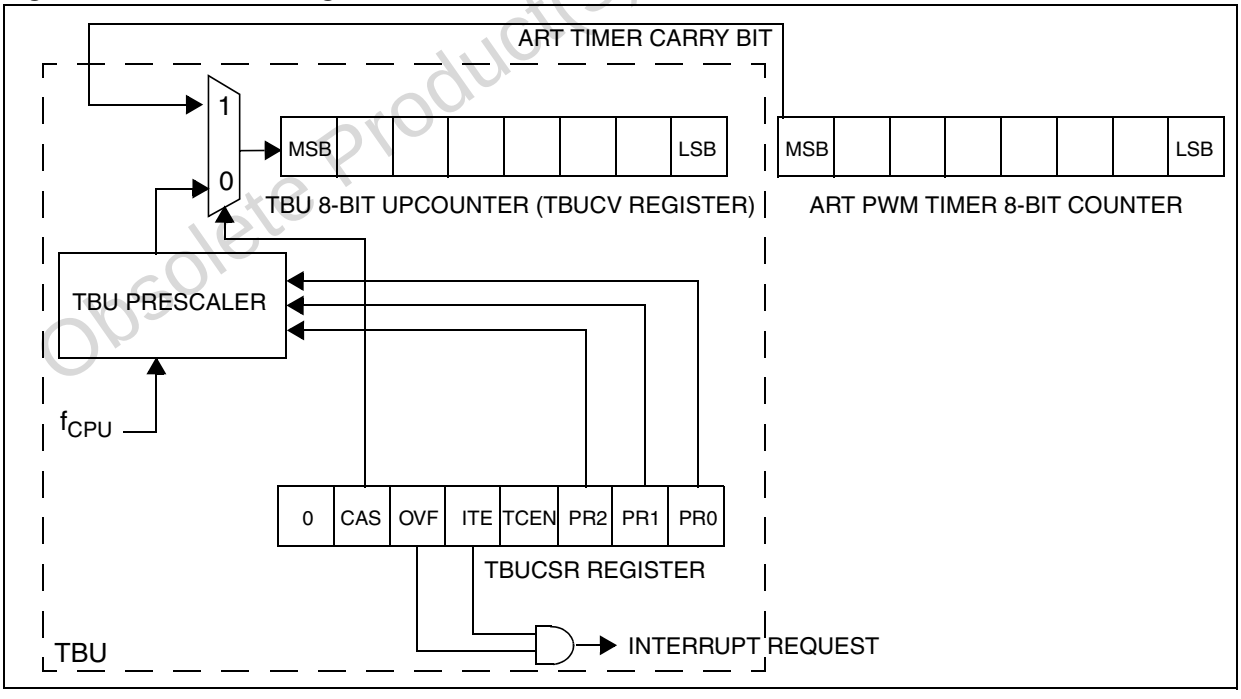
In this example, timer is required to generate an interrupt after a delay of 1 ms.

Assuming that f_{CPU} is 8 MHz and a prescaler division factor of 256 will be programmed using the PR[2:0] bits in the TBUCSR register, 1 ms = 32 TBU timer ticks.

In this case, the initial value to be loaded in the TBUCV must be (256-32) = 224 (E0h).

```
ld A, E0h
ld TBUCV, A   ; Initialize counter value
ld A 1Fh      ;
ld TBUCSR, A  ; Prescaler factor = 256,
              ; interrupt enable,
              ; TBU enable
```

Figure 38. TBU Block Diagram



TIMEBASE UNIT (Cont'd)

10.3.5 Low Power Modes

Mode	Description
WAIT	No effect on TBU
HALT	TBU halted.

10.3.6 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Counter Overflow Event	OVF	ITE	Yes	No

Note: The OVF interrupt event is connected to an interrupt vector (see Interrupts chapter).
It generates an interrupt if the ITE bit is set in the TBUCSR register and the I-bit in the CC register is reset (RIM instruction).

10.3.7 Register Description

TBU COUNTER VALUE REGISTER (TBUCV)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
CV7	CV6	CV5	CV4	CV3	CV2	CV1	CV0

Bit 7:0 = **CV[7:0]** *Counter Value*

This register contains the 8-bit counter value which can be read and written anytime by software. It is continuously incremented by hardware if TCEN=1.

TBU CONTROL/STATUS REGISTER (TBUCSR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	CAS	OVF	ITE	TCEN	PR2	PR1	PR0

Bit 7 = Reserved. Forced by hardware to 0.

Bit 6 = **CAS** *Cascading Enable*
This bit is set and cleared by software. It is used to cascade the TBU and the PWM/ART timers.
0: Cascading disabled
1: Cascading enabled

Bit 5 = **OVF** *Overflow Flag*

This bit is set only by hardware, when the counter value rolls over from FFh to 00h. It is cleared by software reading the TBUCSR register. Writing to this bit does not change the bit value.
0: No overflow
1: Counter overflow

Bit 4 = **ITE** *Interrupt enabled.*

This bit is set and cleared by software.
0: Overflow interrupt disabled
1: Overflow interrupt enabled. An interrupt request is generated when OVF=1.

Bit 3 = **TCEN** *TBU Enable.*

This bit is set and cleared by software.
0: TBU counter is frozen and the prescaler is reset.
1: TBU counter and prescaler running.

Bit 2:0 = **PR[2:0]** *Prescaler Selection*

These bits are set and cleared by software to select the prescaling factor.

PR2	PR1	PR0	Prescaler Division Factor
0	0	0	2
0	0	1	4
0	1	0	8
0	1	1	16
1	0	0	32
1	0	1	64
1	1	0	128
1	1	1	256

TIMEBASE UNIT (Cont'd)

Table 17. TBU Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0036h	**TBUCV** Reset Value	CV7 0	CV6 0	CV5 0	CV4 0	CV3 0	CV2 0	CV1 0	CV0 0
0037h	**TBUSR** Reset Value	- 0	CAS 0	OVF 0	ITE 0	TCEN 0	PR2 0	PR1 0	PR0 0

10.4 SERIAL PERIPHERAL INTERFACE (SPI)

10.4.1 Introduction

The Serial Peripheral Interface (SPI) allows full-duplex, synchronous, serial communication with external devices. An SPI system may consist of a master and one or more slaves however the SPI interface can not be a master in a multimaster system.

10.4.2 Main Features

- Full duplex synchronous transfers (on 3 lines)
- Simplex synchronous transfers (on 2 lines)
- Master or slave operation
- Six master mode frequencies (f_{CPU}/4 max.)
- f_{CPU}/2 max. slave mode frequency (see note)
- $\overline{SS}$ Management by software or hardware
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision, Master Mode Fault and Overrun flags

Note: In slave mode, continuous transmission is not possible at maximum frequency due to the software overhead for clearing status flags and to initiate the next transmission sequence.

10.4.3 General Description

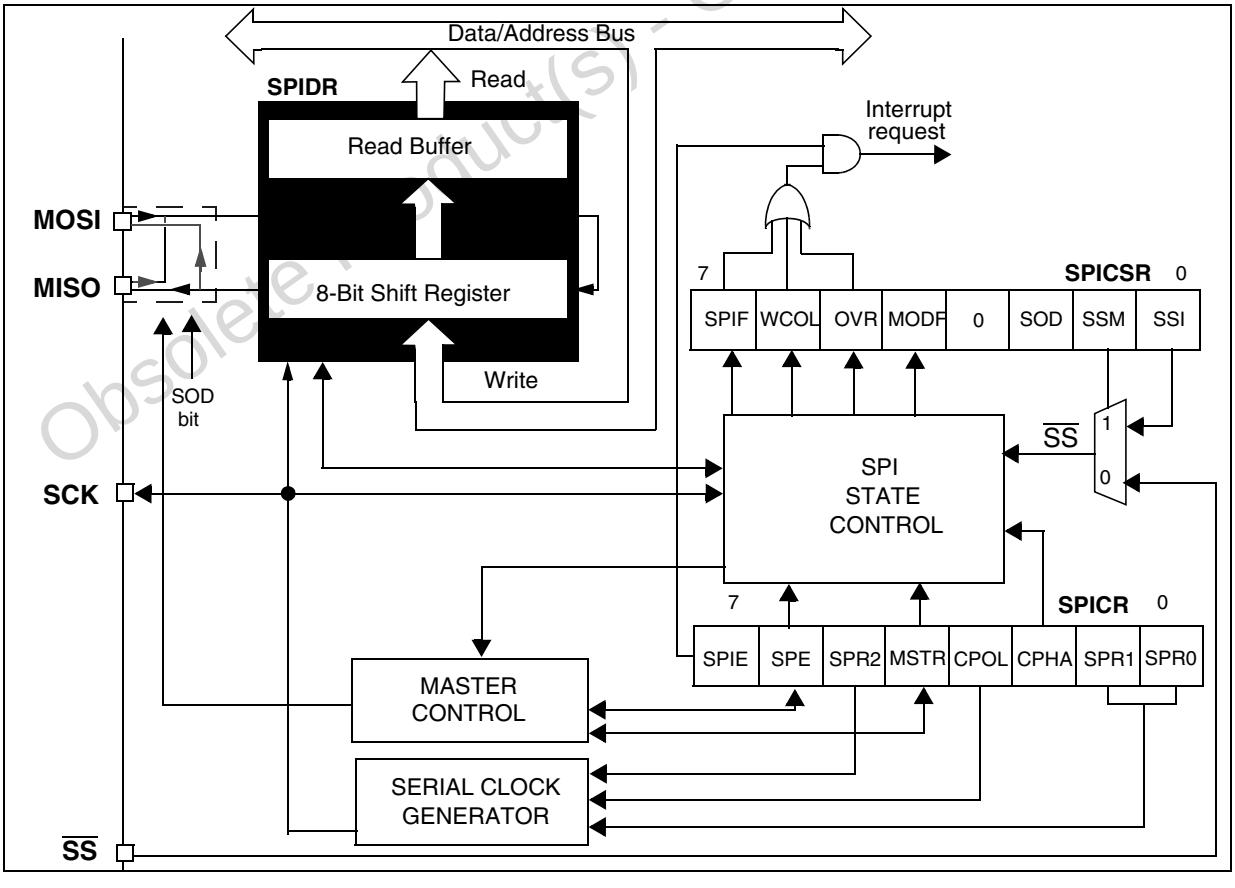
Figure 39 shows the serial peripheral interface (SPI) block diagram. There are 3 registers:

- SPI Control Register (SPICR)
- SPI Control/Status Register (SPICSR)
- SPI Data Register (SPIDR)

The SPI is connected to external devices through 3 pins:

- MISO: Master In / Slave Out data
- MOSI: Master Out / Slave In data
- SCK: Serial Clock out by SPI masters and input by SPI slaves
- $\overline{SS}$: Slave select:
  This input signal acts as a 'chip select' to let the SPI master communicate with slaves individually and to avoid contention on the data lines. Slave $\overline{SS}$ inputs can be driven by standard I/O ports on the master MCU.

Figure 39. Serial Peripheral Interface Block Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

10.4.3.1 Functional Description

A basic example of interconnections between a single master and a single slave is illustrated in Figure 40.

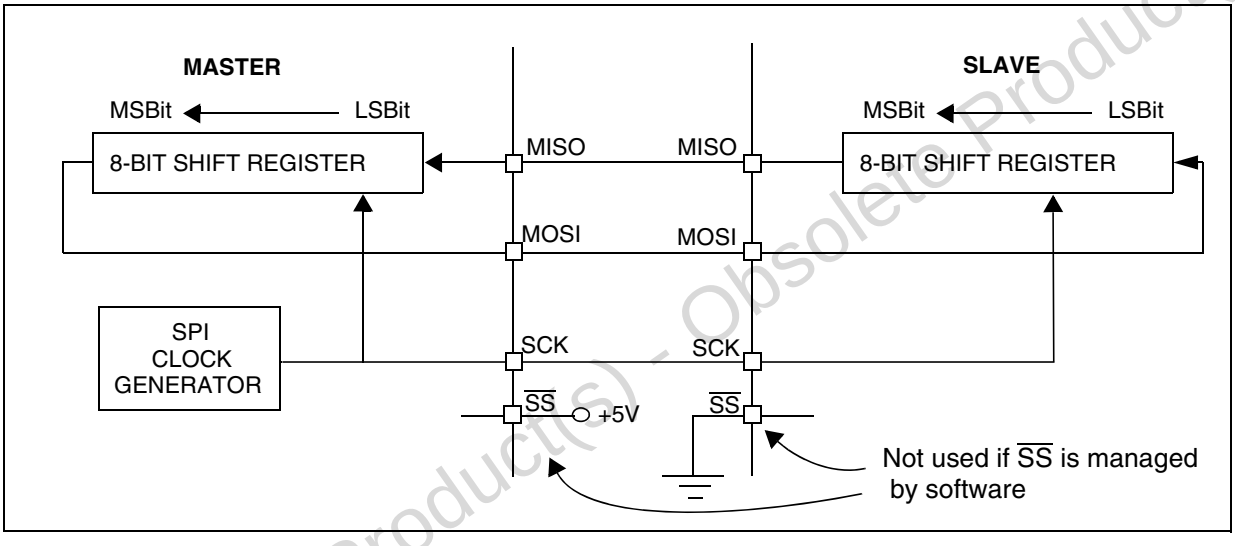
The MOSI pins are connected together and the MISO pins are connected together. In this way data is transferred serially between master and slave (most significant bit first).

The communication is always initiated by the master. When the master device transmits data to a slave device via MOSI pin, the slave device responds by sending data to the master device via the MISO pin. This implies full duplex communication with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

To use a single data line, the MISO and MOSI pins must be connected at each node (in this case only simplex communication is possible).

Four possible data/clock timing relationships may be chosen (see Figure 43) but master and slave must be programmed with the same timing mode.

Figure 40. Single Master/ Single Slave Application



SERIAL PERIPHERAL INTERFACE (Cont'd)

10.4.3.2 Slave Select Management

As an alternative to using the $\overline{SS}$ pin to control the Slave Select signal, the application can choose to manage the Slave Select signal by software. This is configured by the SSM bit in the SPICSR register (see Figure 42)

In software management, the external $\overline{SS}$ pin is free for other application uses and the internal $\overline{SS}$ signal level is driven by writing to the SSI bit in the SPICSR register.

In Master mode:

– $\overline{SS}$ internal must be held high continuously

In Slave Mode:

There are two cases depending on the data/clock timing relationship (see Figure 41):

If CPHA=1 (data latched on 2nd clock edge):

– $\overline{SS}$ internal must be held low during the entire transmission. This implies that in single slave applications the $\overline{SS}$ pin either can be tied to V_{SS}, or made free for standard I/O by managing the $\overline{SS}$ function by software (SSM= 1 and SSI=0 in the in the SPICSR register)

If CPHA=0 (data latched on 1st clock edge):

– $\overline{SS}$ internal must be held low during byte transmission and pulled high between each byte to allow the slave to write to the shift register. If $\overline{SS}$ is not pulled high, a Write Collision error will occur when the slave writes to the shift register (see Section 10.4.5.3).

Figure 41. Generic $\overline{SS}$ Timing Diagram

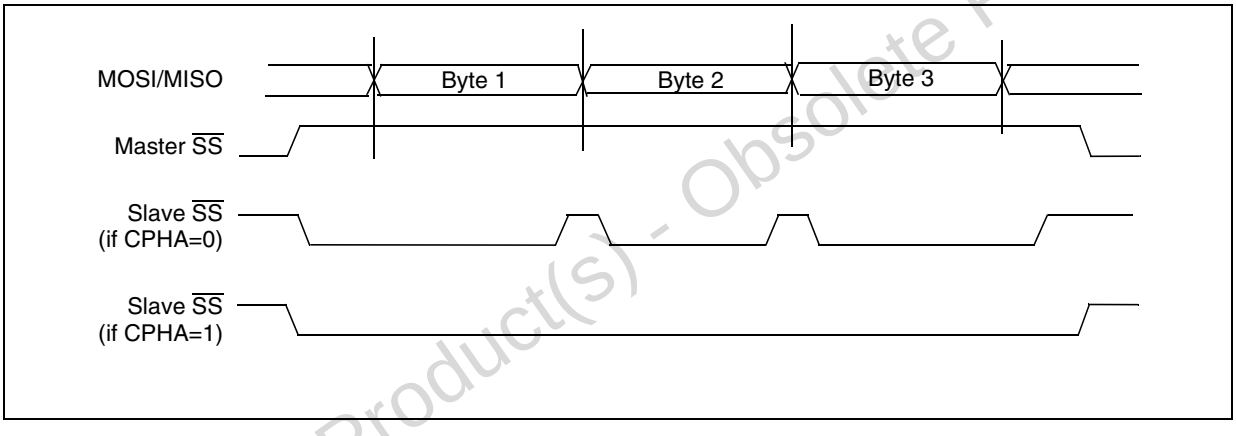
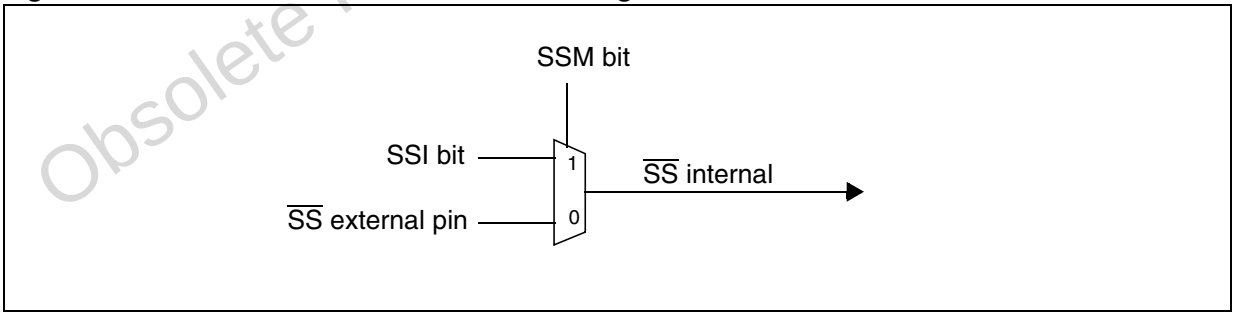


Figure 42. Hardware/Software Slave Select Management



SERIAL PERIPHERAL INTERFACE (Cont'd)

10.4.3.3 Master Mode Operation

In master mode, the serial clock is output on the SCK pin. The clock frequency, polarity and phase are configured by software (refer to the description of the SPICSR register).

Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL=1 or pulling down SCK if CPOL=0).

To operate the SPI in master mode, perform the following two steps in order (if the SPICSR register is not written first, the SPICR register setting (MSTR bit) may be not taken into account):

1. Write to the SPICR register:
   - Select the clock frequency by configuring the SPR[2:0] bits.
   - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits. Figure 43 shows the four possible configurations.
     Note: The slave must have the same CPOL and CPHA settings as the master.
2. Write to the SPICSR register:
   - Either set the SSM bit and set the SSI bit or clear the SSM bit and tie the SS pin high for the complete byte transmit sequence.
3. Write to the SPICR register:
   - Set the MSTR and SPE bits
     Note: MSTR and SPE bits remain set only if SS is high).

The transmit sequence begins when software writes a byte in the SPIDR register.

10.4.3.4 Master Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MOSI pin most significant bit first.

When data transfer is complete:

- The SPIF bit is set by hardware
- An interrupt request is generated if the SPIE bit is set and the interrupt mask in the CC register is cleared.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SPICSR register while the SPIF bit is set
2. A read to the SPIDR register.

Note: While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

10.4.3.5 Slave Mode Operation

In slave mode, the serial clock is received on the SCK pin from the master device.

To operate the SPI in slave mode:

1. Write to the SPICSR register to perform the following actions:
   - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits (see Figure 43).
     Note: The slave must have the same CPOL and CPHA settings as the master.
   - Manage the $\overline{SS}$ pin as described in Section 10.4.3.2 and Figure 41. If CPHA=1 $\overline{SS}$ must be held low continuously. If CPHA=0 $\overline{SS}$ must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
2. Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

10.4.3.6 Slave Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MISO pin most significant bit first.

The transmit sequence begins when the slave device receives the clock signal and the most significant bit of the data on its MOSI pin.

When data transfer is complete:

- The SPIF bit is set by hardware
- An interrupt request is generated if SPIE bit is set and interrupt mask in the CC register is cleared.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SPICSR register while the SPIF bit is set.
2. A write or a read to the SPIDR register.

Notes: While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

The SPIF bit can be cleared during a second transmission; however, it must be cleared before the second SPIF bit in order to prevent an Overrun condition (see Section 10.4.5.2).

SERIAL PERIPHERAL INTERFACE (Cont'd)

10.4.4 Clock Phase and Clock Polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (See Figure 43).

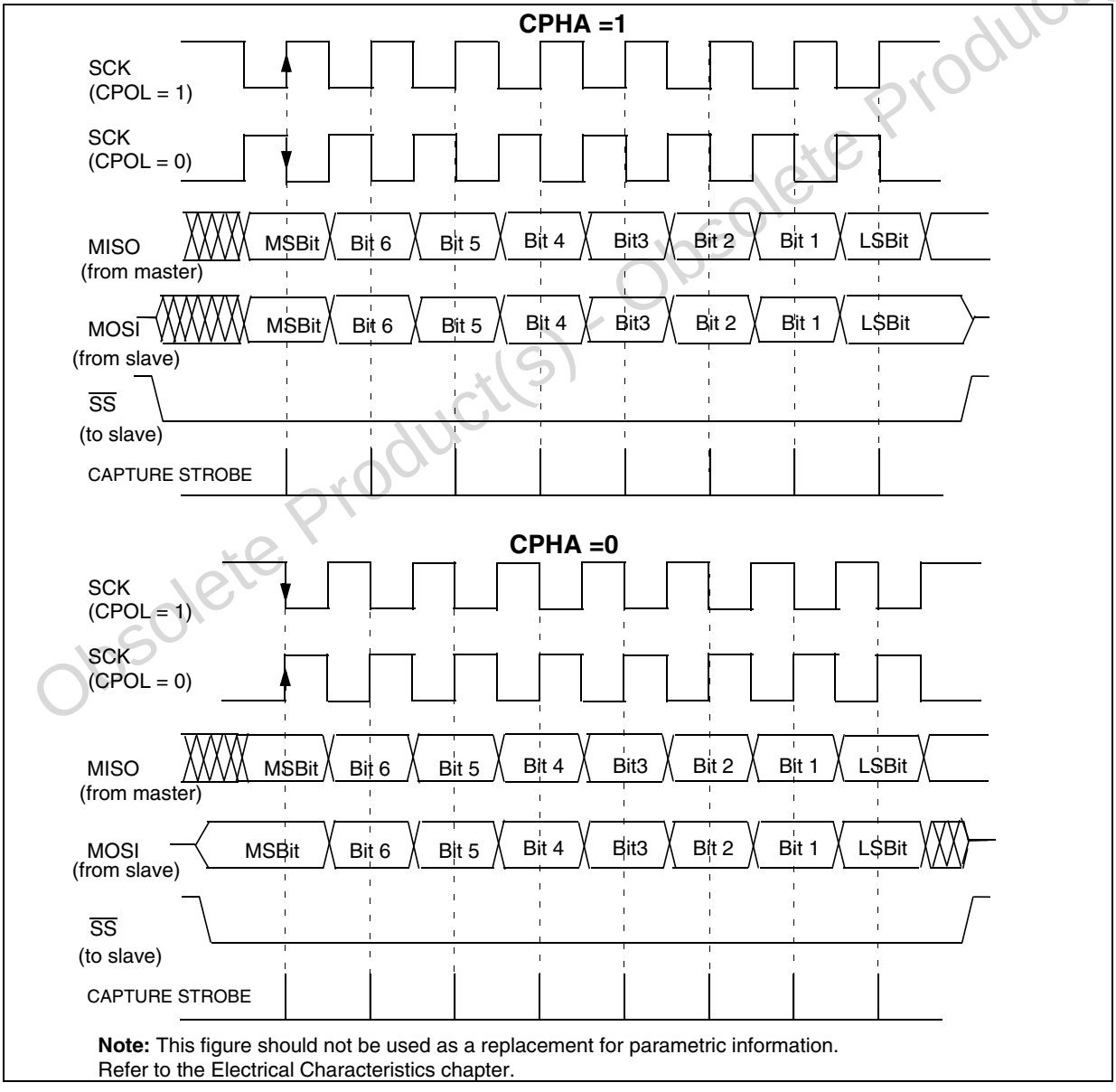
Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL=1 or pulling down SCK if CPOL=0).

The combination of the CPOL clock polarity and CPHA (clock phase) bits selects the data capture clock edge

Figure 43, shows an SPI transfer with the four combinations of the CPHA and CPOL bits. The diagram may be interpreted as a master or slave timing diagram where the SCK pin, the MISO pin, the MOSI pin are directly connected between the master and the slave device.

Note: If CPOL is changed at the communication byte boundaries, the SPI must be disabled by resetting the SPE bit.

Figure 43. Data Clock Timing Diagram



Note: This figure should not be used as a replacement for parametric information. Refer to the Electrical Characteristics chapter.

SERIAL PERIPHERAL INTERFACE (Cont'd)

10.4.5 Error Flags

10.4.5.1 Master Mode Fault (MODF)

Master mode fault occurs when the master device has its $\overline{SS}$ pin pulled low.

When a Master mode fault occurs:

– The MODF bit is set and an SPI interrupt request is generated if the SPIE bit is set.

– The SPE bit is reset. This blocks all output from the device and disables the SPI peripheral.

– The MSTR bit is reset, thus forcing the device into slave mode.

Clearing the MODF bit is done through a software sequence:

1. A read access to the SPICSR register while the MODF bit is set.

2. A write to the SPICR register.

Notes: To avoid any conflicts in an application with multiple slaves, the $\overline{SS}$ pin must be pulled high during the MODF bit clearing sequence. The SPE and MSTR bits may be restored to their original state during or after this clearing sequence.

Hardware does not allow the user to set the SPE and MSTR bits while the MODF bit is set except in the MODF bit clearing sequence.

In a slave device, the MODF bit can not be set, but in a multimaster configuration the device can be in slave mode with the MODF bit set.

The MODF bit indicates that there might have been a multimaster conflict and allows software to handle this using an interrupt routine and either perform to a reset or return to an application default state.

10.4.5.2 Overrun Condition (OVR)

An overrun condition occurs, when the master device has sent a data byte and the slave device has not cleared the SPIF bit issued from the previously transmitted byte.

When an Overrun occurs:

– The OVR bit is set and an interrupt request is generated if the SPIE bit is set.

In this case, the receiver buffer contains the byte sent after the SPIF bit was last cleared. A read to the SPIDR register returns this byte. All other bytes are lost.

The OVR bit is cleared by reading the SPICSR register.

10.4.5.3 Write Collision Error (WCOL)

A write collision occurs when the software tries to write to the SPIDR register while a data transfer is taking place with an external device. When this happens, the transfer continues uninterrupted; and the software write will be unsuccessful.

Write collisions can occur both in master and slave mode. See also Section 10.4.3.2 Slave Select Management.

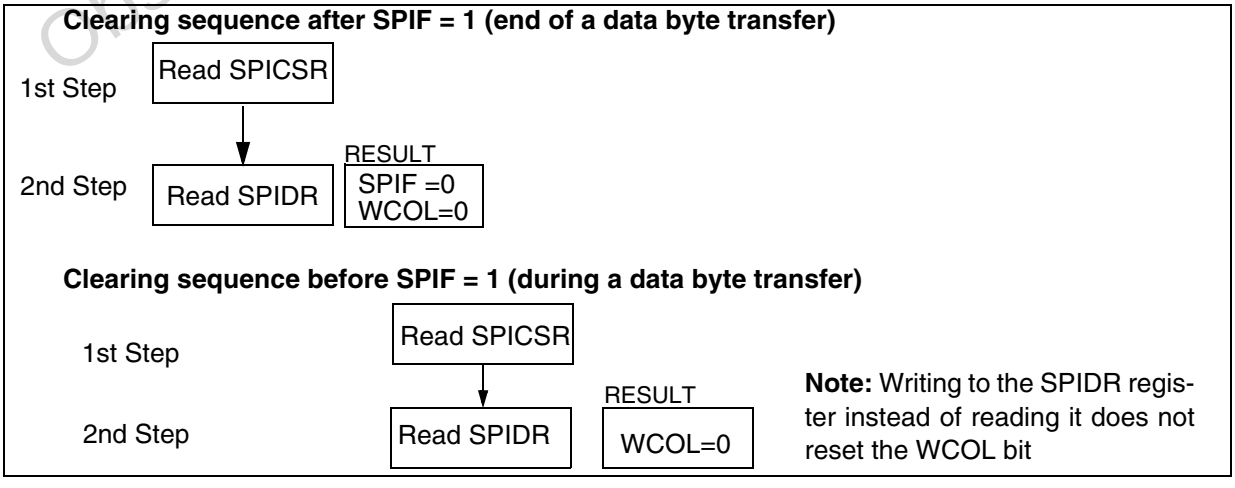
Note: a "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the MCU operation.

The WCOL bit in the SPICSR register is set if a write collision occurs.

No SPI interrupt is generated when the WCOL bit is set (the WCOL bit is a status flag only).

Clearing the WCOL bit is done through a software sequence (see Figure 44).

Figure 44. Clearing the WCOL bit (Write Collision Flag) Software Sequence



SERIAL PERIPHERAL INTERFACE (Cont'd)

10.4.5.4 Single Master System

A typical single master system may be configured, using an MCU as the master and four MCUs as slaves (see Figure 45).

The master device selects the individual slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the slave devices.

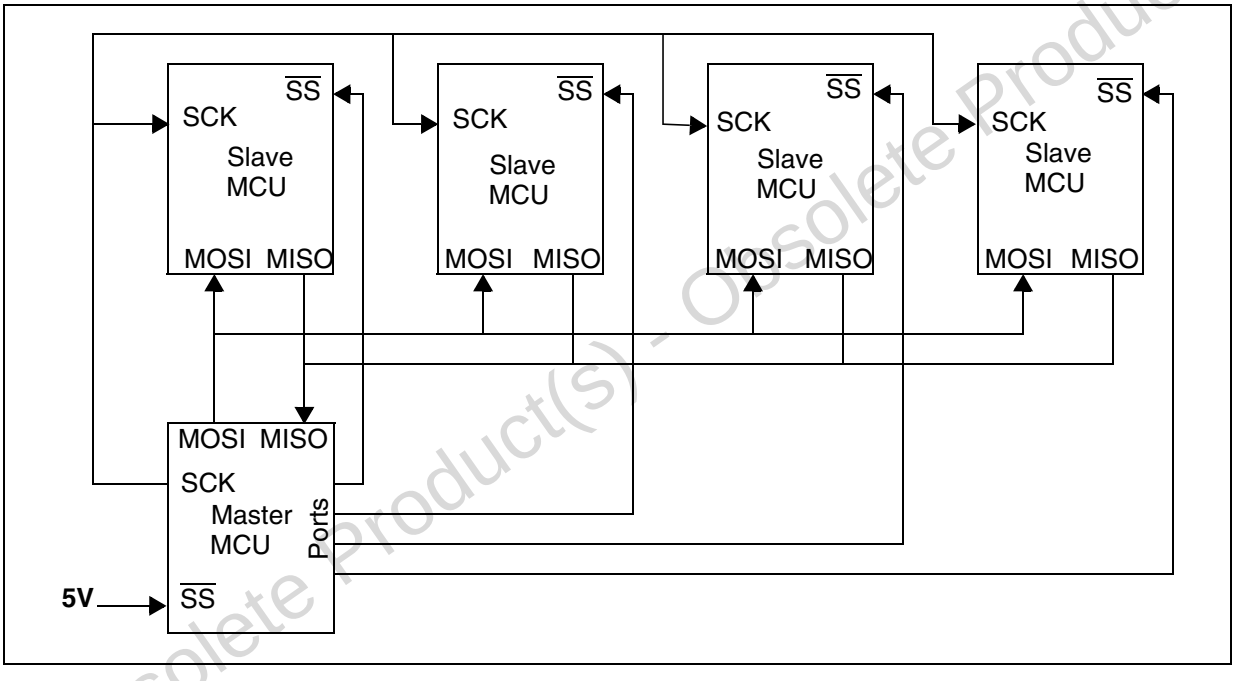
The $\overline{SS}$ pins are pulled high during reset since the master device ports will be forced to be inputs at that time, thus disabling the slave devices.

Note: To prevent a bus conflict on the MISO line the master allows only one active slave device during a transmission.

For more security, the slave device may respond to the master with the received data byte. Then the master will receive the previous byte back from the slave device if all MISO and MOSI pins are connected and the slave has not written to its SPIDR register.

Other transmission security methods can use ports for handshake lines or data bytes with command fields.

Figure 45. Single Master / Multiple Slave Configuration



SERIAL PERIPHERAL INTERFACE (Cont'd)

10.4.6 Low Power Modes

Mode	Description
WAIT	No effect on SPI. SPI interrupt events cause the device to exit from WAIT mode.
HALT	SPI registers are frozen. In HALT mode, the SPI is inactive. SPI operation resumes when the MCU is woken up by an interrupt with "exit from HALT mode" capability. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetching). If several data are received before the wake-up event, then an overrun error is generated. This error can be detected after the fetch of the interrupt routine that woke up the device.

10.4.6.1 Using the SPI to wakeup the MCU from Halt mode

In slave configuration, the SPI is able to wakeup the ST7 device from HALT mode through a SPIF interrupt. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetch). If multiple data transfers have been performed before software clears the SPIF bit, then the OVR bit is set by hardware.

Note: When waking up from Halt mode, if the SPI remains in Slave mode, it is recommended to perform an extra communications cycle to bring the SPI from Halt mode state to normal state. If the SPI exits from Slave mode, it returns to normal state immediately.

Caution: The SPI can wake up the ST7 from Halt mode only if the Slave Select signal (external $\overline{SS}$ pin or the SSI bit in the SPICSR register) is low when the ST7 enters Halt mode. So if Slave selection is configured as external (see Section 10.4.3.2), make sure the master drives a low level on the $\overline{SS}$ pin when the slave enters Halt mode.

10.4.7 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
SPI End of Transfer Event	SPIF	SPIE	Yes	Yes
Master Mode Fault Event	MODF		Yes	No
Overrun Error	OVR		Yes	No

Note: The SPI interrupt events are connected to the same interrupt vector (see Interrupts chapter). They generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in

SERIAL PERIPHERAL INTERFACE (Cont'd)

10.4.8 Register Description

CONTROL REGISTER (SPICR)

Read/Write

Reset Value: 0000 xxxx (0xh)

7							0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0

Bit 7 = **SPIE** *Serial Peripheral Interrupt Enable.*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SPI interrupt is generated whenever SPIF=1, MODF=1 or OVR=1 in the SPICSR register

Bit 6 = **SPE** *Serial Peripheral Output Enable.*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$=0 (see Section 10.4.5.1 Master Mode Fault (MODF)). The SPE bit is cleared by reset, so the SPI peripheral is not initially connected to the external pins.
0: I/O pins free for general purpose I/O
1: SPI I/O pin alternate functions enabled

Bit 5 = **SPR2** *Divider Enable*.
This bit is set and cleared by software and is cleared by reset. It is used with the SPR[1:0] bits to set the baud rate. Refer to Table 18 SPI Master mode SCK Frequency.
0: Divider by 2 enabled
1: Divider by 2 disabled

Note: This bit has no effect in slave mode.

Bit 4 = **MSTR** *Master Mode.*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$=0 (see Section 10.4.5.1 Master Mode Fault (MODF)).
0: Slave mode
1: Master mode. The function of the SCK pin changes from an input to an output and the functions of the MISO and MOSI pins are reversed.

Bit 3 = **CPOL** *Clock Polarity.*
This bit is set and cleared by software. This bit determines the idle state of the serial Clock. The CPOL bit affects both the master and slave modes.
0: SCK pin has a low level idle state
1: SCK pin has a high level idle state

Note: If CPOL is changed at the communication byte boundaries, the SPI must be disabled by resetting the SPE bit.

Bit 2 = **CPHA** *Clock Phase.*
This bit is set and cleared by software.
0: The first clock transition is the first data capture edge.
1: The second clock transition is the first capture edge.

Note: The slave must have the same CPOL and CPHA settings as the master.

Bits 1:0 = **SPR[1:0]** *Serial Clock Frequency.*
These bits are set and cleared by software. Used with the SPR2 bit, they select the baud rate of the SPI serial clock SCK output by the SPI in master mode.

Note: These 2 bits have no effect in slave mode.

Table 18. SPI Master mode SCK Frequency

SPR2	SPR1	SPR0	Serial Clock (f_{CPU} = 8MHz)	Serial Clock (f_{CPU}= 4MHz)	SCK
1	0	0	$f_{CPU}/4$	$f_{CPU}/2$	2 MHz
0	0	0	$f_{CPU}/8$	$f_{CPU}/4$	1 MHz
0	0	1	$f_{CPU}/16$	$f_{CPU}/8$	0.5 MHz
1	1	0	$f_{CPU}/32$	$f_{CPU}/16$	0.25 MHz
0	1	0	$f_{CPU}/64$	$f_{CPU}/32$	125 kHz
0	1	1	$f_{CPU}/128$	$f_{CPU}/64$	62.5 kHz

SERIAL PERIPHERAL INTERFACE (Cont'd)

CONTROL/STATUS REGISTER (SPICSR)

Read/Write (some bits Read Only)
Reset Value: 0000 0000 (00h)

7							0
SPIF	WCOL	OVR	MODF	-	SOD	SSM	SSI

Bit 7 = **SPIF** *Serial Peripheral Data Transfer Flag (Read only).*
This bit is set by hardware when a transfer has been completed. An interrupt is generated if SPIE=1 in the SPICR register. It is cleared by a software sequence (an access to the SPICSR register followed by a write or a read to the SPIDR register).
0: Data transfer is in progress or the flag has been cleared.
1: Data transfer between the device and an external device has been completed.

Note: While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

Bit 6 = **WCOL** *Write Collision status (Read only).*
This bit is set by hardware when a write to the SPIDR register is done during a transmit sequence. It is cleared by a software sequence (see Figure 44).
0: No write collision occurred
1: A write collision has been detected

Bit 5 = **OVR** *SPI Overrun error (Read only).*
This bit is set by hardware when the byte currently being received in the shift register is ready to be transferred into the SPIDR register while SPIF = 1 (See Section 10.4.5.2). An interrupt is generated if SPIE = 1 in SPICR register. The OVR bit is cleared by software reading the SPICSR register.
0: No overrun error
1: Overrun error detected

Bit 4 = **MODF** *Mode Fault flag (Read only).*
This bit is set by hardware when the $\overline{SS}$ pin is pulled low in master mode (see Section 10.4.5.1 Master Mode Fault (MODF)). An SPI interrupt can be generated if SPIE=1 in the SPICR register. This bit is cleared by a software sequence (An access to the SPICSR register while MODF=1 followed by a write to the SPICR register).
0: No master mode fault detected
1: A fault in master mode has been detected

Bit 3 = Reserved, must be kept cleared.

Bit 2 = **SOD** *SPI Output Disable.*
This bit is set and cleared by software. When set, it disables the alternate function of the SPI output (MOSI in master mode / MISO in slave mode)
0: SPI output enabled (if SPE=1)
1: SPI output disabled

Bit 1 = **SSM** $\overline{SS}$ *Management.*
This bit is set and cleared by software. When set, it disables the alternate function of the SPI $\overline{SS}$ pin and uses the SSI bit value instead. See Section 10.4.3.2 Slave Select Management.
0: Hardware management ($\overline{SS}$ managed by external pin)
1: Software management (internal $\overline{SS}$ signal controlled by SSI bit. External $\overline{SS}$ pin free for general-purpose I/O)

Bit 0 = **SSI** $\overline{SS}$ *Internal Mode.*
This bit is set and cleared by software. It acts as a 'chip select' by controlling the level of the $\overline{SS}$ slave select signal when the SSM bit is set.
0 : Slave selected
1 : Slave deselected

DATA I/O REGISTER (SPIDR)

Read/Write
Reset Value: Undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0

The SPIDR register is used to transmit and receive data on the serial bus. In a master device, a write to this register will initiate transmission/reception of another byte.

Notes: During the last clock cycle the SPIF bit is set, a copy of the received data byte in the shift register is moved to a buffer. When the user reads the serial peripheral data I/O register, the buffer is actually being read.

While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

Warning: A write to the SPIDR register places data directly into the shift register for transmission.

A read to the SPIDR register returns the value located in the buffer and not the content of the shift register (see Figure 39).

Table 19. SPI Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0011h	**SPIDR** Reset Value	MSB x	x	x	x	x	x	x	LSB x
0012h	**SPICR** Reset Value	SPIE 0	SPE 0	SPR2 0	MSTR 0	CPOL x	CPHA x	SPR1 x	SPR0 x
0013h	**SPICSR** Reset Value	SPIF 0	WCOL 0	OVR 0	MODF 0	0	SOD 0	SSM 0	SSI 0

10.5 SERIAL COMMUNICATIONS INTERFACE (SCI)

10.5.1 Introduction

The Serial Communications Interface (SCI) offers a flexible means of full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The SCI offers a very wide range of baud rates using two baud rate generator systems.

10.5.2 Main Features

- Full duplex, asynchronous communications
- NRZ standard format (Mark/Space)
- Dual baud rate generator systems
- Independently programmable transmit and receive baud rates up to 500K baud
- Programmable data word length (8 or 9 bits)
- Receive buffer full, Transmit buffer empty and End of Transmission flags
- Two receiver wake-up modes:
  - Address bit (MSB)
  - Idle line
- Muting function for multiprocessor configurations
- Separate enable bits for Transmitter and Receiver
- Four error detection flags:
  - Overrun error
  - Noise error
  - Frame error
  - Parity error
- Five interrupt sources with flags:
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line received
  - Overrun error detected
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Reduced power consumption mode

10.5.3 General Description

The interface is externally connected to another device by two pins (see Figure 47):

– TDO: Transmit Data Output. When the transmitter and the receiver are disabled, the output pin returns to its I/O port configuration. When the transmitter and/or the receiver are enabled and nothing is to be transmitted, the TDO pin is at high level.

– RDI: Receive Data Input is the serial data input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

Through these pins, serial data is transmitted and received as frames comprising:

– An Idle Line prior to transmission or reception

– A start bit

– A data word (8 or 9 bits) least significant bit first

– A Stop bit indicating that the frame is complete

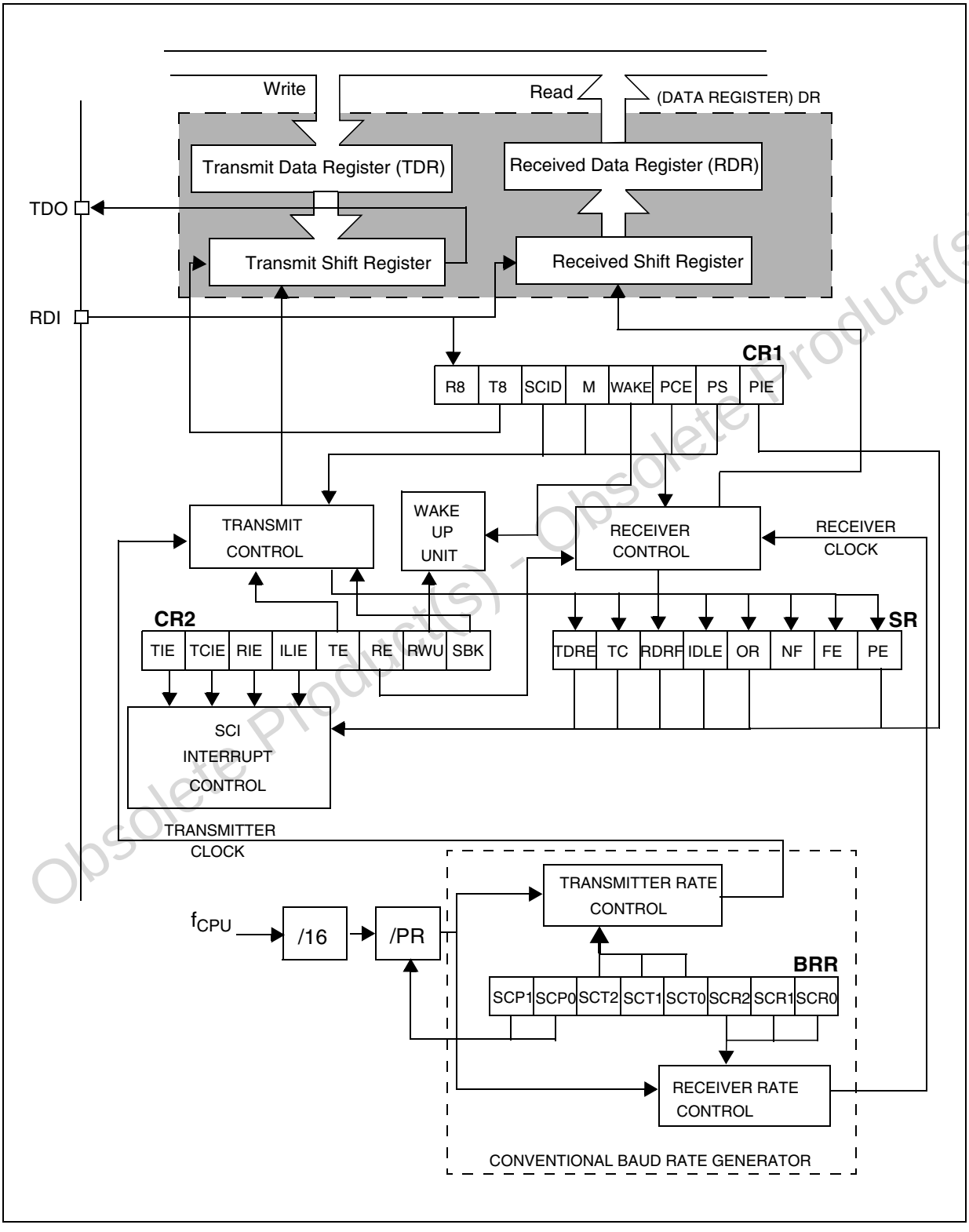
This interface uses two types of baud rate generator:

– A conventional type for commonly-used baud rates

– An extended type with a prescaler offering a very wide range of baud rates even with non-standard oscillator frequencies

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 46. SCI Block Diagram



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.4 Functional Description

The block diagram of the Serial Control Interface, is shown in Figure 46 It contains six dedicated registers:

– Two control registers (SCICR1 & SCICR2)

– A status register (SCISR)

– A baud rate register (SCIBRR)

– An extended prescaler receiver register (SCIERPR)

– An extended prescaler transmitter register (SCIETPR)

Refer to the register descriptions in Section 10.5.7for the definitions of each bit.

10.5.4.1 Serial Data Format

Word length may be selected as being either 8 or 9 bits by programming the M bit in the SCICR1 register (see Figure 46).

The TDO pin is in low state during the start bit.

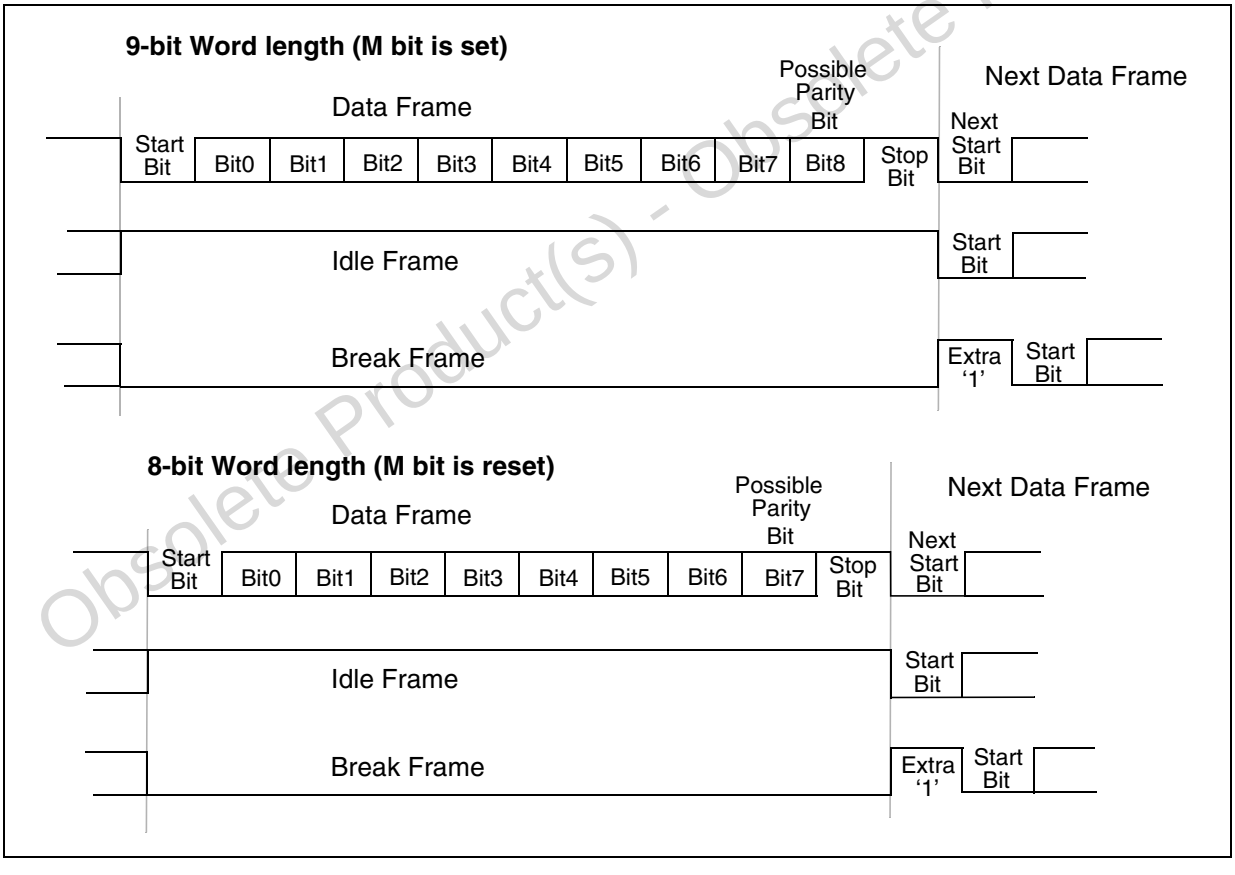
The TDO pin is in high state during the stop bit.

An Idle character is interpreted as an entire frame of "1"s followed by the start bit of the next frame which contains data.

A Break character is interpreted on receiving "0"s for some multiple of the frame period. At the end of the last break frame the transmitter inserts an extra "1" bit to acknowledge the start bit.

Transmission and reception are driven by their own baud rate generator.

Figure 47. Word Length Programming



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.4.2 Transmitter

The transmitter can send data words of either 8 or 9 bits depending on the M bit status. When the M bit is set, word length is 9 bits and the 9th bit (the MSB) has to be stored in the T8 bit in the SCICR1 register.

Character Transmission

During an SCI transmission, data shifts out least significant bit first on the TDO pin. In this mode, the SCIDR register consists of a buffer (TDR) between the internal bus and the transmit shift register (see Figure 46).

Procedure

– Select the M bit to define the word length.

– Select the desired baud rate using the SCIBRR and the SCIETPR registers.

– Set the TE bit to assign the TDO pin to the alternate function and to send a idle frame as first transmission.

– Access the SCISR register and write the data to send in the SCIDR register (this sequence clears the TDRE bit). Repeat this sequence for each data to be transmitted.

Clearing the TDRE bit is always performed by the following software sequence:

1. An access to the SCISR register

2. A write to the SCIDR register

The TDRE bit is set by hardware and it indicates:

– The TDR register is empty.

– The data transfer is beginning.

– The next data can be written in the SCIDR register without overwriting the previous data.

This flag generates an interrupt if the TIE bit is set and the I bit is cleared in the CC register.

When a transmission is taking place, a write instruction to the SCIDR register stores the data in the TDR register and which is copied in the shift register at the end of the current transmission.

When no transmission is taking place, a write instruction to the SCIDR register places the data directly in the shift register, the data transmission starts, and the TDRE bit is immediately set.

When a frame transmission is complete (after the stop bit) the TC bit is set and an interrupt is generated if the TCIE is set and the I bit is cleared in the CC register.

Clearing the TC bit is performed by the following software sequence:

1. An access to the SCISR register

2. A write to the SCIDR register

Note: The TDRE and TC bits are cleared by the same software sequence.

Break Characters

Setting the SBK bit loads the shift register with a break character. The break frame length depends on the M bit (see Figure 47).

As long as the SBK bit is set, the SCI send break frames to the TDO pin. After clearing this bit by software the SCI insert a logic 1 bit at the end of the last break frame to guarantee the recognition of the start bit of the next frame.

Idle Characters

Setting the TE bit drives the SCI to send an idle frame before the first data frame.

Clearing and then setting the TE bit during a transmission sends an idle frame after the current word.

Note: Resetting and setting the TE bit causes the data in the TDR register to be lost. Therefore the best time to toggle the TE bit is when the TDRE bit is set, that is, before writing the next byte in the SCIDR.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.4.3 Receiver

The SCI can receive data words of either 8 or 9 bits. When the M bit is set, word length is 9 bits and the MSB is stored in the R8 bit in the SCICR1 register.

Character reception

During a SCI reception, data shifts in least significant bit first through the RDI pin. In this mode, the SCIDR register consists or a buffer (RDR) between the internal bus and the received shift register (see Figure 46).

Procedure

– Select the M bit to define the word length.

– Select the desired baud rate using the SCIBRR and the SCIERPR registers.

– Set the RE bit, this enables the receiver which begins searching for a start bit.

When a character is received:

– The RDRF bit is set. It indicates that the content of the shift register is transferred to the RDR.

– An interrupt is generated if the RIE bit is set and the I bit is cleared in the CC register.

– The error flags can be set if a frame error, noise or an overrun error has been detected during reception.

Clearing the RDRF bit is performed by the following software sequence done by:

1. An access to the SCISR register

2. A read to the SCIDR register.

The RDRF bit must be cleared before the end of the reception of the next character to avoid an overrun error.

Break Character

When a break character is received, the SCI handles it as a framing error.

Idle Character

When a idle frame is detected, there is the same procedure as a data received character plus an interrupt if the ILIE bit is set and the I bit is cleared in the CC register.

Overrun Error

An overrun error occurs when a character is received when RDRF has not been reset. Data can not be transferred from the shift register to the RDR register as long as the RDRF bit is not cleared.

When an overrun error occurs:

– The OR bit is set.

– The RDR content is not lost.

– The shift register is overwritten.

– An interrupt is generated if the RIE bit is set and the I bit is cleared in the CC register.

The OR bit is reset by an access to the SCISR register followed by a SCIDR register read operation.

Noise Error

Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise. Normal data bits are considered valid if three consecutive samples (8th, 9th, 10th) have the same bit value, otherwise the NF flag is set. In the case of start bit detection, the NF flag is set on the basis of an algorithm combining both valid edge detection and three samples (8th, 9th, 10th). Therefore, to prevent the NF flag getting set during start bit reception, there should be a valid edge detection as well as three valid samples.

When noise is detected in a frame:

– The NF flag is set at the rising edge of the RDRF bit.

– Data is transferred from the Shift register to the SCIDR register.

– No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The NF flag is reset by a SCISR register read operation followed by a SCIDR register read operation.

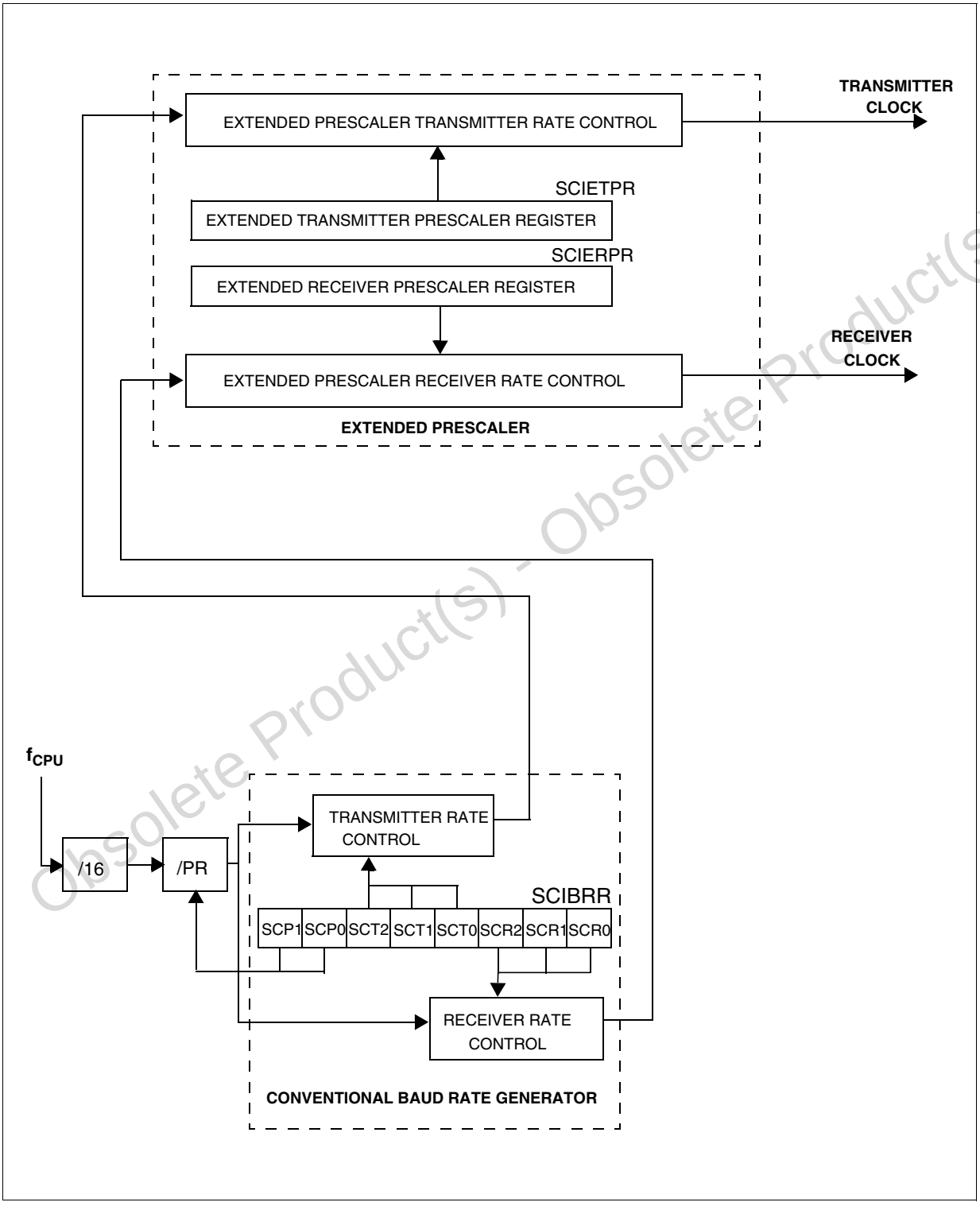
During reception, if a false start bit is detected (e.g. 8th, 9th, 10th samples are 011,101,110), the frame is discarded and the receiving sequence is not started for this frame. There is no RDRF bit set for this frame and the NF flag is set internally (not accessible to the user). This NF flag is accessible along with the RDRF bit when a next valid frame is received.

Note: If the application Start Bit is not long enough to match the above requirements, then the NF Flag may get set due to the short Start Bit. In this case, the NF flag may be ignored by the application software when the first valid byte is received.

See also Section 10.5.4.10.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 48. SCI Baud Rate and Extended Prescaler Block Diagram



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Framing Error

A framing error is detected when:

– The stop bit is not recognized on reception at the expected time, following either a de-synchronization or excessive noise.

– A break is received.

When the framing error is detected:

– the FE bit is set by hardware

– Data is transferred from the Shift register to the SCIDR register.

– No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The FE bit is reset by a SCISR register read operation followed by a SCIDR register read operation.

10.5.4.4 Conventional Baud Rate Generation

The baud rate for the receiver and transmitter (Rx and Tx) are set independently and calculated as follows:

$$Tx = \frac{f_{CPU}}{(16*PR)*TR} \qquad Rx = \frac{f_{CPU}}{(16*PR)*RR}$$

with:

PR = 1, 3, 4 or 13 (see SCP[1:0] bits)

TR = 1, 2, 4, 8, 16, 32, 64,128

(see SCT[2:0] bits)

RR = 1, 2, 4, 8, 16, 32, 64,128

(see SCR[2:0] bits)

All these bits are in the SCIBRR register.

Example: If f_{CPU} is 8 MHz (normal mode) and if PR = 13 and TR = RR = 1, the transmit and receive baud rates are 38400 baud.

Note: The baud rate registers MUST NOT be changed while the transmitter or the receiver is enabled.

10.5.4.5 Extended Baud Rate Generation

The extended prescaler option gives a very fine tuning on the baud rate, using a 255 value prescaler, whereas the conventional Baud Rate Generator retains industry standard software compatibility.

The extended baud rate generator block diagram is described in the Figure 48

The output clock rate sent to the transmitter or to the receiver is the output from the 16 divider divided by a factor ranging from 1 to 255 set in the SCIERPR or the SCIETPR register.

Note: the extended prescaler is activated by setting the SCIETPR or SCIERPR register to a value other than zero. The baud rates are calculated as follows:

$$Tx = \frac{f_{CPU}}{16*ETPR*(PR*TR)} \qquad Rx = \frac{f_{CPU}}{16*ERPR*(PR*RR)}$$

with:

ETPR = 1,..,255 (see SCIETPR register)

ERPR = 1,.. 255 (see SCIERPR register)

10.5.4.6 Receiver Muting and Wake-up Feature

In multiprocessor configurations it is often desirable that only the intended message recipient should actively receive the full message contents, thus reducing redundant SCI service overhead for all non addressed receivers.

The non addressed devices may be placed in sleep mode by means of the muting function.

Setting the RWU bit by software puts the SCI in sleep mode:

All the reception status bits can not be set.

All the receive interrupts are inhibited.

A muted receiver may be awakened by one of the following two ways:

– by Idle Line detection if the WAKE bit is reset,

– by Address Mark detection if the WAKE bit is set.

Receiver wakes-up by Idle Line detection when the Receive line has recognized an Idle Frame. Then the RWU bit is reset by hardware but the IDLE bit is not set.

Receiver wakes-up by Address Mark detection when it received a "1" as the most significant bit of a word, thus indicating that the message is an address. The reception of this particular word wakes up the receiver, resets the RWU bit and sets the RDRF bit, which allows the receiver to receive this word normally and to use it as an address word.

CAUTION: In Mute mode, do not write to the SCICR2 register. If the SCI is in Mute mode during the read operation (RWU = 1) and a address mark wake up event occurs (RWU is reset) before the write operation, the RWU bit is set again by this write operation. Consequently the address byte is lost and the SCI is not woken up from Mute mode.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.4.7 Parity Control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PCE bit in the SCICR1 register. Depending on the frame length defined by the M bit, the possible SCI frame formats are as listed in Table 20.

Table 20. Frame Formats

M bit	PCE bit	SCI frame
0	0	| SB | 8 bit data | STB |
0	1	| SB | 7-bit data | PB | STB |
1	0	| SB | 9-bit data | STB |
1	1	| SB | 8-bit data PB | STB |

Legend: SB = Start Bit, STB = Stop Bit, PB = Parity Bit

Note: In case of wake up by an address mark, the MSB bit of the data is taken into account and not the parity bit

Even parity: the parity bit is calculated to obtain an even number of "1s" inside the frame made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit is 0 if even parity is selected (PS bit = 0).

Odd parity: the parity bit is calculated to obtain an odd number of "1s" inside the frame made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit is 1 if odd parity is selected (PS bit = 1).

Transmission mode: If the PCE bit is set then the MSB bit of the data written in the data register is not transmitted but is changed by the parity bit.

Reception mode: If the PCE bit is set then the interface checks if the received data byte has an even number of "1s" if even parity is selected (PS = 0) or an odd number of "1s" if odd parity is selected (PS = 1). If the parity check fails, the PE flag is set in the SCISR register and an interrupt is generated if PIE is set in the SCICR1 register.

10.5.4.8 SCI Clock Tolerance

During reception, each bit is sampled 16 times. The majority of the 8th, 9th and 10th samples is considered as the bit value. For a valid bit detection, all the three samples should have the same value otherwise the noise flag (NF) is set. For example: If the 8th, 9th and 10th samples are 0, 1 and 1 respectively, then the bit value is "1", but the Noise Flag bit is set because the three samples values are not the same.

Consequently, the bit length must be long enough so that the 8th, 9th and 10th samples have the desired bit value. This means the clock frequency should not vary more than 6/16 (37.5%) within one bit. The sampling clock is resynchronized at each start bit, so that when receiving 10 bits (one start bit, 1 data byte, 1 stop bit), the clock deviation must not exceed 3.75%.

Note: The internal sampling clock of the microcontroller samples the pin value on every falling edge. Therefore, the internal sampling clock and the time the application expects the sampling to take place may be out of sync. For example: If the baud rate is 15.625 Kbaud (bit length is 64µs), then the 8th, 9th and 10th samples are at 28µs, 32µs and 36µs respectively (the first sample starting ideally at 0µs). But if the falling edge of the internal clock occurs just before the pin value changes, the samples would then be out of sync by ~4us. This means the entire bit length must be at least 40µs (36µs for the 10th sample + 4µs for synchronization with the internal sampling clock).

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.4.9 Clock Deviation Causes

The causes which contribute to the total deviation are:

- D_{TRA}: Deviation due to transmitter error (Local oscillator error of the transmitter or the transmitter is transmitting at a different baud rate).
- D_{QUANT}: Error due to the baud rate quantization of the receiver.
- D_{REC}: Deviation of the local oscillator of the receiver: This deviation can occur during the reception of one complete SCI message assuming that the deviation has been compensated at the beginning of the message.
- D_{TCL}: Deviation due to the transmission line (generally due to the transceivers)

All the deviations of the system should be added and compared to the SCI clock tolerance:

$D_{TRA} + D_{QUANT} + D_{REC} + D_{TCL} < 3.75\%$

10.5.4.10 Noise Error Causes

See also description of Noise error in Section 10.5.4.3.

Start bit

The noise flag (NF) is set during start bit reception if one of the following conditions occurs:

1. A valid falling edge is not detected. A falling edge is considered to be valid if the 3 consecutive samples before the falling edge occurs are detected as '1' and, after the falling edge occurs, during the sampling of the 16 samples, if one of the samples numbered 3, 5 or 7 is detected as a "1".
2. During sampling of the 16 samples, if one of the samples numbered 8, 9 or 10 is detected as a "1".

Therefore, a valid Start Bit must satisfy both the above conditions to prevent the Noise Flag getting set.

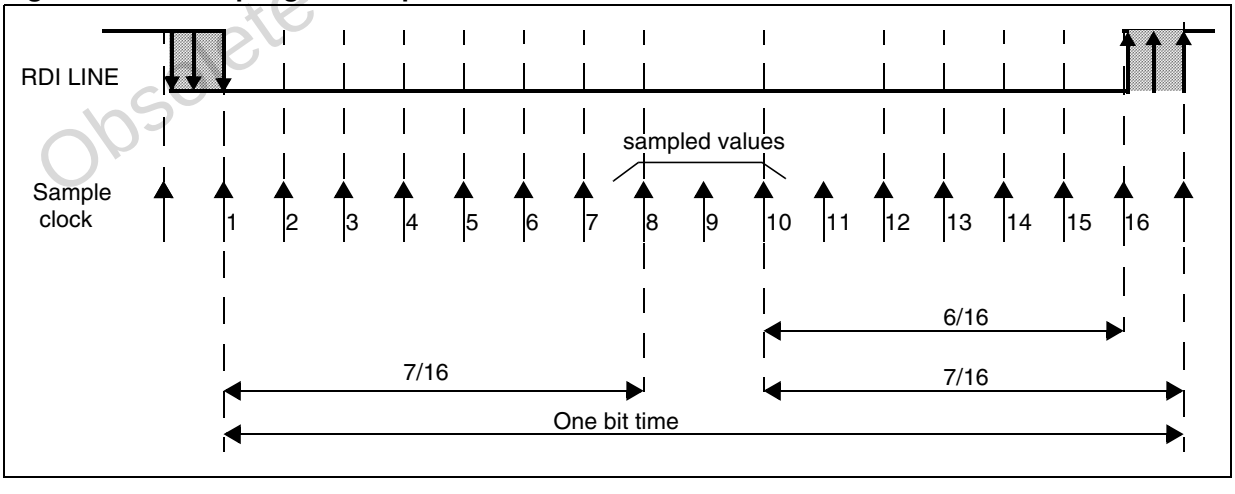
Data Bits

The noise flag (NF) is set during normal data bit reception if the following condition occurs:

- During the sampling of 16 samples, if all three samples numbered 8, 9 and10 are not the same. The majority of the 8th, 9th and 10th samples is considered as the bit value.

Therefore, a valid Data Bit must have samples 8, 9 and 10 at the same value to prevent the Noise Flag getting set.

Figure 49. Bit Sampling in Reception Mode



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.5 Low Power Modes

Mode	Description
WAIT	No effect on SCI. SCI interrupts cause the device to exit from Wait mode.
HALT	SCI registers are frozen. In Halt mode, the SCI stops transmitting/receiving until Halt mode is exited.

10.5.6 Interrupts

The SCI interrupt events are connected to the same interrupt vector.

These events generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in the CC register is reset (RIM instruction).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Transmit Data Register Empty	TDRE	TIE	Yes	No
Transmission Complete	TC	TCIE	Yes	No
Received Data Ready to be Read	RDRF	RIE	Yes	No
Overrun Error Detected	OR		Yes	No
Idle Line Detected	IDLE	ILIE	Yes	No
Parity Error	PE	PIE	Yes	No

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

10.5.7 Register Description

STATUS REGISTER (SCISR)

Read Only

Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	OR	NF	FE	PE

Bit 7 = **TDRE** *Transmit data register empty.*
This bit is set by hardware when the content of the TDR register has been transferred into the shift register. An interrupt is generated if the TIE bit = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).
0: Data is not transferred to the shift register
1: Data is transferred to the shift register

Note: Data is not transferred to the shift register unless the TDRE bit is cleared.

Bit 6 = **TC** *Transmission complete.*
This bit is set by hardware when transmission of a frame containing Data is complete. An interrupt is generated if TCIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).
0: Transmission is not complete
1: Transmission is complete

Note: TC is not set after the transmission of a Preamble or a Break.

Bit 5 = **RDRF** *Received data ready flag.*
This bit is set by hardware when the content of the RDR register has been transferred to the SCIDR register. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: Data is not received
1: Received data is ready to be read

Bit 4 = **IDLE** *Idle line detect.*
This bit is set by hardware when a Idle Line is detected. An interrupt is generated if the ILIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Idle Line is detected
1: Idle Line is detected

Note: The IDLE bit is not set again until the RDRF bit has been set itself (that is, a new idle line occurs).

Bit 3 = **OR** *Overrun error.*
This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RDRF = 1. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Overrun error
1: Overrun error is detected

Note: When this bit is set RDR register content is not lost but the shift register is overwritten.

Bit 2 = **NF** *Noise flag.*
This bit is set by hardware when noise is detected on a received frame. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No noise is detected
1: Noise is detected

Note: This bit does not generate interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt.

Bit 1 = **FE** *Framing error.*
This bit is set by hardware when a de-synchronization, excessive noise or a break character is detected. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Framing error is detected
1: Framing error or break character is detected

Note: This bit does not generate interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt. If the word currently being transferred causes both frame error and overrun error, it will be transferred and only the OR bit will be set.

Bit 0 = **PE** *Parity error.*
This bit is set by hardware when a parity error occurs in receiver mode. It is cleared by a software sequence (a read to the status register followed by an access to the SCIDR data register). An interrupt is generated if PIE = 1 in the SCICR1 register.
0: No parity error
1: Parity error

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

CONTROL REGISTER 1 (SCICR1)

Read/Write

Reset Value: x000 0000 (x0h)

7							0
R8	T8	SCID	M	WAKE	PCE	PS	PIE

Bit 7 = **R8** *Receive data bit 8.*
This bit is used to store the 9th bit of the received word when M = 1.

Bit 6 = **T8** *Transmit data bit 8.*
This bit is used to store the 9th bit of the transmitted word when M = 1.

Bit 5 = **SCID** *Disabled for low power consumption*
When this bit is set the SCI prescalers and outputs are stopped and the end of the current byte transfer in order to reduce power consumption.This bit is set and cleared by software.
0: SCI enabled
1: SCI prescaler and outputs disabled

Bit 4 = **M** *Word length.*
This bit determines the word length. It is set or cleared by software.
0: 1 Start bit, 8 Data bits, 1 Stop bit
1: 1 Start bit, 9 Data bits, 1 Stop bit

Note: The M bit must not be modified during a data transfer (both transmission and reception).

Bit 3 = **WAKE** *Wake-Up method.*
This bit determines the SCI Wake-Up method, it is set or cleared by software.
0: Idle Line
1: Address Mark

Bit 2 = **PCE** *Parity control enable.*
This bit selects the hardware parity control (generation and detection). When the parity control is enabled, the computed parity is inserted at the MSB position (9th bit if M = 1; 8th bit if M = 0) and parity is checked on the received data. This bit is set and cleared by software. Once it is set, PCE is active after the current byte (in reception and in transmission).
0: Parity control disabled
1: Parity control enabled

Bit 1 = **PS** *Parity selection.*
This bit selects the odd or even parity when the parity generation/detection is enabled (PCE bit set). It is set and cleared by software. The parity is selected after the current byte.
0: Even parity
1: Odd parity

Bit 0 = **PIE** *Parity interrupt enable.*
This bit enables the interrupt capability of the hardware parity control when a parity error is detected (PE bit set). It is set and cleared by software.
0: Parity error interrupt disabled
1: Parity error interrupt enabled.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

CONTROL REGISTER 2 (SCICR2)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

Bit 7 = **TIE** *Transmitter interrupt enable*.
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever TDRE=1 in the SCISR register

Bit 6 = TCIE *Transmission complete interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever TC=1 in the SCISR register

Bit 5 = **RIE** *Receiver interrupt enable*.
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever OR=1 or RDRF=1 in the SCISR register

Bit 4 = **ILIE** *Idle line interrupt enable.*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever IDLE=1 in the SCISR register.

Bit 3 = **TE** *Transmitter enable.*
This bit enables the transmitter. It is set and cleared by software.
0: Transmitter is disabled
1: Transmitter is enabled

Notes:

– During transmission, a "0" pulse on the TE bit ("0" followed by "1") sends a preamble (idle line) after the current word.

– When TE is set there is a 1 bit-time delay before the transmission starts.

CAUTION: The TDO pin is free for general purpose I/O only when the TE and RE bits are both cleared (or if TE is never set).

Bit 2 = **RE** *Receiver enable.*
This bit enables the receiver. It is set and cleared by software.
0: Receiver is disabled
1: Receiver is enabled and begins searching for a start bit

Bit 1 = **RWU** *Receiver wake-up.*
This bit determines if the SCI is in mute mode or not. It is set and cleared by software and can be cleared by hardware when a wake-up sequence is recognized.
0: Receiver in Active mode
1: Receiver in Mute mode

Note: Before selecting Mute mode (setting the RWU bit), the SCI must receive some data first, otherwise it cannot function in Mute mode with wake-up by idle line detection.

Bit 0 = **SBK** *Send break.*
This bit set is used to send break characters. It is set and cleared by software.
0: No break character is transmitted
1: Break characters are transmitted

Note: If the SBK bit is set to "1" and then to "0", the transmitter sends a BREAK word at the end of the current word.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

DATA REGISTER (SCIDR)

Read/Write

Reset Value: Undefined

Contains the Received or Transmitted data character, depending on whether it is read from or written to.

7							0
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0

The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR).
The TDR register provides the parallel interface between the internal bus and the output shift register (see Figure 46).
The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 46).

BAUD RATE REGISTER (SCIBRR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0

Bits 7:6 = **SCP[1:0]** *First SCI Prescaler*
These 2 prescaling bits allow several standard clock division ranges:

PR Prescaling factor	SCP1	SCP0
1	0	0
3	0	1
4	1	0
13	1	1

Bits 5:3 = **SCT[2:0]** *SCI Transmitter rate divisor*
These 3 bits, in conjunction with the SCP1 & SCP0 bits define the total division applied to the bus clock to yield the transmit rate clock in conventional Baud Rate Generator mode.

TR dividing factor	SCT2	SCT1	SCT0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

Bits 2:0 = **SCR[2:0]** *SCI Receiver rate divisor.*
These 3 bits, in conjunction with the SCP[1:0] bits define the total division applied to the bus clock to yield the receive rate clock in conventional Baud Rate Generator mode.

RR Dividing factor	SCR2	SCR1	SCR0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

EXTENDED RECEIVE PRESCALER DIVISION REGISTER (SCIERPR)

Read/Write

Reset Value: 0000 0000 (00h)

Allows setting of the Extended Prescaler rate division factor for the receive circuit.

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

Bits 7:0 = **ERPR[7:0]** *8-bit Extended Receive Prescaler Register.*

The extended Baud Rate Generator is activated when a value different from 00h is stored in this register. Therefore the clock frequency issued from the 16 divider (see Figure 48) is divided by the binary factor set in the SCIERPR register (in the range 1 to 255).

The extended baud rate generator is not used after a reset.

EXTENDED TRANSMIT PRESCALER DIVISION REGISTER (SCIETPR)

Read/Write

Reset Value:0000 0000 (00h)

Allows setting of the External Prescaler rate division factor for the transmit circuit.

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

Bits 7:0 = **ETPR[7:0]** *8-bit Extended Transmit Prescaler Register.*

The extended Baud Rate Generator is activated when a value different from 00h is stored in this register. Therefore the clock frequency issued from the 16 divider (see Figure 48) is divided by the binary factor set in the SCIETPR register (in the range 1 to 255).

The extended baud rate generator is not used after a reset.

Table 21. Baudrate Selection

Symbol	Parameter	Conditions			Standard	Baud Rate	Unit
		f_{CPU}	Accuracy vs Standard	Prescaler			
f_{Tx} f_{Rx}	Communication frequency	8 MHz	~0.16%	Conventional Mode TR (or RR)=128, PR=13 TR (or RR)= 32, PR=13 TR (or RR)= 16, PR=13 TR (or RR)= 8, PR=13 TR (or RR)= 4, PR=13 TR (or RR)= 16, PR= 3 TR (or RR)= 2, PR=13 TR (or RR)= 1, PR=13	 300 1200 2400 4800 9600 10400 19200 38400	 ~300.48 ~1201.92 ~2403.84 ~4807.69 ~9615.38 ~10416.67 ~19230.77 ~38461.54	Hz
			~0.79%	Extended Mode ETPR (or ERPR) = 35, TR (or RR)= 1, PR=1	14400	~14285.71	

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Table 22. SCI Register Map and Reset Values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
1D	**SCIERPR**	ERPR7	ERPR6	ERPR5	ERPR4	ERPR3	ERPR2	ERPR1	ERPR0
	Reset Value	0	0	0	0	0	0	0	0
1E	**SCIETPR**	ETPR7	ETPR6	ETPR5	ETPR4	ETPR3	ETPR2	ETPR1	ETPR0
	Reset Value	0	0	0	0	0	0	0	0
20	**SCISR**	TDRE	TC	RDRF	IDLE	OR	NF	FE	PE
	Reset Value	1	1	0	0	0	0	0	0
21	**SCIDR**	DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0
	Reset Value	x	x	x	x	x	x	x	x
22	**SCIBRR**	SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0
	Reset Value	0	0	0	0	0	0	0	0
23	**SCICR1**	R8	T8	SCID	M	WAKE	PCE	PS	PIE
	Reset Value	x	0	0	0	0	0	0	0
24	**SCICR2**	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
	Reset Value	0	0	0	0	0	0	0	0

10.6 USB INTERFACE (USB)

10.6.1 Introduction

The USB Interface implements a low-speed function interface between the USB and the ST7 microcontroller. It is a highly integrated circuit which includes the transceiver, 3.3 voltage regulator, SIE and DMA. No external components are needed apart from the external pull-up on USBDM for low speed recognition by the USB host. The use of DMA architecture allows the endpoint definition to be completely flexible. Endpoints can be configured by software as in or out.

10.6.2 Main Features

- USB Specification Version 1.1 Compliant
- Supports Low-Speed USB Protocol
- Two or Three Endpoints (including default one) depending on the device (see device feature list and register map)
- CRC generation/checking, NRZI encoding/decoding and bit-stuffing
- USB Suspend/Resume operations
- DMA Data transfers
- On-Chip 3.3V Regulator
- On-Chip USB Transceiver

10.6.3 Functional Description

The block diagram in Figure 50, gives an overview of the USB interface hardware.

For general information on the USB, refer to the "Universal Serial Bus Specifications" document available at http//:www.usb.org.

Serial Interface Engine

The SIE (Serial Interface Engine) interfaces with the USB, via the transceiver.

The SIE processes tokens, handles data transmission/reception, and handshaking as required by the USB standard. It also performs frame formatting, including CRC generation and checking.

Endpoints

The Endpoint registers indicate if the microcontroller is ready to transmit/receive, and how many bytes need to be transmitted.

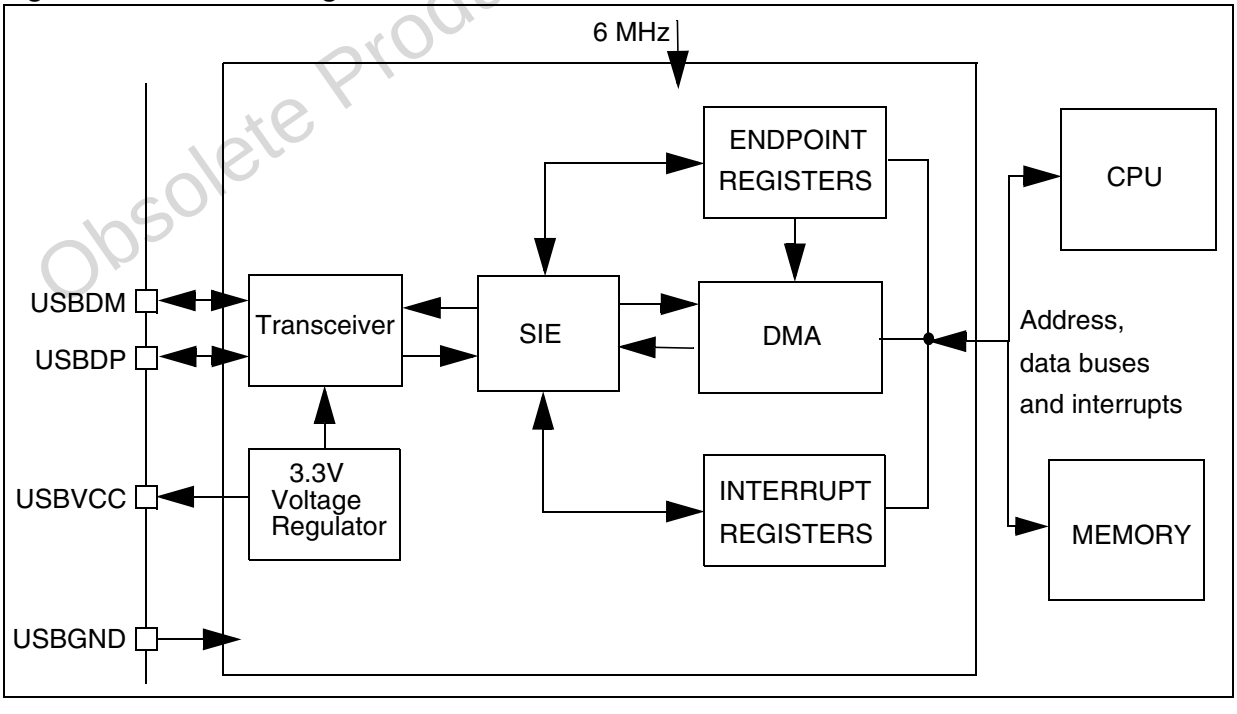
DMA

When a token for a valid Endpoint is recognized by the USB interface, the related data transfer takes place, using DMA. At the end of the transaction, an interrupt is generated.

Interrupts

By reading the Interrupt Status register, application software can know which USB event has occurred.

Figure 50. USB Block Diagram



USB INTERFACE (Cont'd)

10.6.4 Register Description

DMA ADDRESS REGISTER (DMAR)

Read / Write

Reset Value: Undefined

7							0
DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8

Bits 7:0=**DA[15:8]** *DMA address bits 15-8.*
Software must write the start address of the DMA memory area whose most significant bits are given by DA15-DA6. The remaining 6 address bits are set by hardware. See the description of the IDR register and Figure 51.

INTERRUPT/DMA REGISTER (IDR)

Read / Write

Reset Value: xxxx 0000 (x0h)

7							0
DA7	DA6	EP1	EP0	CNT3	CNT2	CNT1	CNT0

Bits 7:6 = **DA[7:6]** *DMA address bits 7-6.*
Software must reset these bits. See the description of the DMAR register and Figure 51.

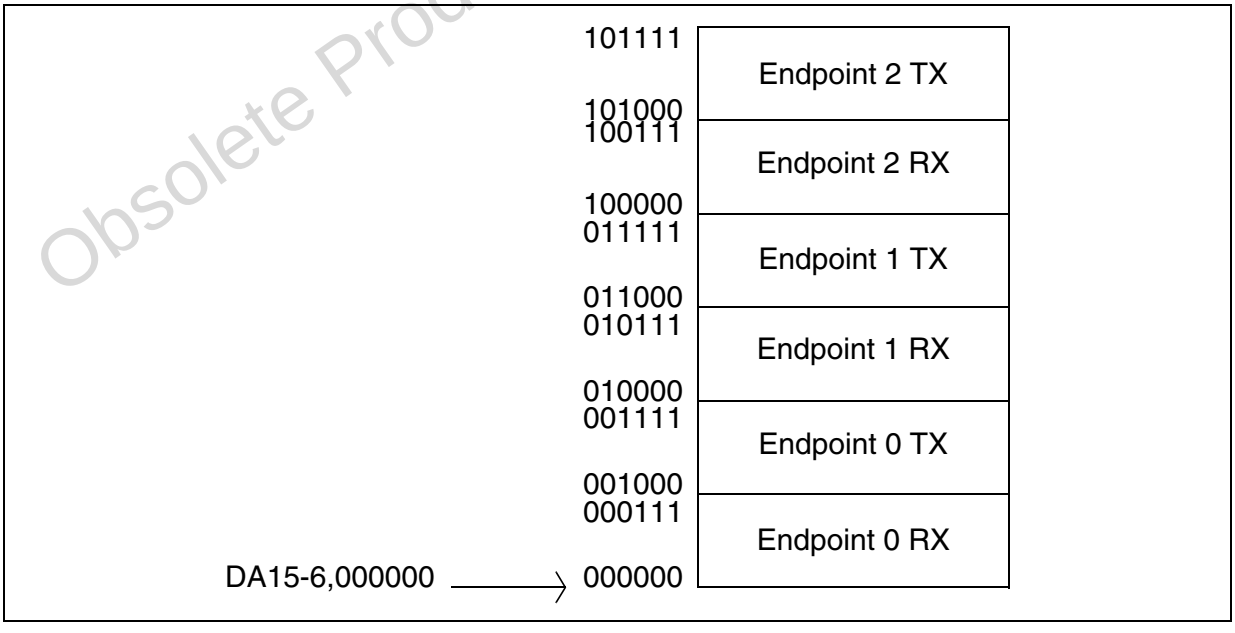
Bits 5:4 = **EP[1:0]** *Endpoint number* (read-only).
These bits identify the endpoint which required attention.
00: Endpoint 0
01: Endpoint 1
10: Endpoint 2

When a CTR interrupt occurs (see register ISTR) the software should read the EP bits to identify the endpoint which has sent or received a packet.

Bits 3:0 = **CNT[3:0]** *Byte count* (read only).
This field shows how many data bytes have been received during the last data reception.

Note: Not valid for data transmission.

Figure 51. DMA Buffers



USB INTERFACE (Cont'd)

PID REGISTER (PIDR)

Read only

Reset Value: xx00 0000 (x0h)

7							0
TP3	TP2	0	0	0	RX_ SEZ	RXD	0

Bits 7:6 = **TP[3:2]** *Token PID bits 3 & 2*.
USB token PIDs are encoded in four bits. **TP[3:2]** correspond to the variable token PID bits 3 & 2.
Note: PID bits 1 & 0 have a fixed value of 01.
When a CTR interrupt occurs (see register ISTR) the software should read the TP3 and TP2 bits to retrieve the PID name of the token received.
The USB standard defines TP bits as:

TP3	TP2	PID Name
0	0	OUT
1	0	IN
1	1	SETUP

Bits 5:3 Reserved. Forced by hardware to 0.

Bit 2 = **RX_SEZ** *Received single-ended zero*
This bit indicates the status of the RX_SEZ transceiver output.
0: No SE0 (single-ended zero) state
1: USB lines are in SE0 (single-ended zero) state

Bit 1 = **RXD** *Received data*
0: No K-state
1: USB lines are in K-state

This bit indicates the status of the RXD transceiver output (differential receiver output).

Note: If the environment is noisy, the RX_SEZ and RXD bits can be used to secure the application. By interpreting the status, software can distinguish a valid End Suspend event from a spurious wake-up due to noise on the external USB line. A valid End Suspend is followed by a Resume or Reset sequence. A Resume is indicated by RXD=1, a Reset is indicated by RX_SEZ=1.

Bit 0 = Reserved. Forced by hardware to 0.

INTERRUPT STATUS REGISTER (ISTR)

Read / Write

Reset Value: 0000 0000 (00h)

7							0
SUSP	DOVR	CTR	ERR	IOVR	ESUSP	RESET	SOF

When an interrupt occurs these bits are set by hardware. Software must read them to determine the interrupt type and clear them after servicing.
Note: These bits cannot be set by software.

Bit 7 = **SUSP** *Suspend mode request*.
This bit is set by hardware when a constant idle state is present on the bus line for more than 3 ms, indicating a suspend mode request from the USB bus. The suspend request check is active immediately after each USB reset event and its disabled by hardware when suspend mode is forced (FSUSP bit of CTLR register) until the end of resume sequence.

Bit 6 = **DOVR** *DMA over/underrun*.
This bit is set by hardware if the ST7 processor can't answer a DMA request in time.
0: No over/underrun detected
1: Over/underrun detected

Bit 5 = **CTR** *Correct Transfer.* This bit is set by hardware when a correct transfer operation is performed. The type of transfer can be determined by looking at bits TP3-TP2 in register PIDR. The Endpoint on which the transfer was made is identified by bits EP1-EP0 in register IDR.
0: No Correct Transfer detected
1: Correct Transfer detected

Note: A transfer where the device sent a NAK or STALL handshake is considered not correct (the host only sends ACK handshakes). A transfer is considered correct if there are no errors in the PID and CRC fields, if the DATA0/DATA1 PID is sent as expected, if there were no data overruns, bit stuffing or framing errors.

Bit 4 = **ERR** *Error.*
This bit is set by hardware whenever one of the errors listed below has occurred:
0: No error detected
1: Timeout, CRC, bit stuffing or nonstandard framing error detected

USB INTERFACE (Cont'd)

Bit 3 = **IOVR** *Interrupt overrun.*
This bit is set when hardware tries to set ERR, or SOF before they have been cleared by software.
0: No overrun detected
1: Overrun detected

Bit 2 = **ESUSP** *End suspend mode*.
This bit is set by hardware when, during suspend mode, activity is detected that wakes the USB interface up from suspend mode.

This interrupt is serviced by a specific vector, in order to wake up the ST7 from HALT mode.
0: No End Suspend detected
1: End Suspend detected

Bit 1 = **RESET** *USB reset.*
This bit is set by hardware when the USB reset sequence is detected on the bus.
0: No USB reset signal detected
1: USB reset signal detected

Note: The DADDR, EP0RA, EP0RB, EP1RA, EP1RB, EP2RA and EP2RB registers are reset by a USB reset.

Bit 0 = **SOF** *Start of frame.*
This bit is set by hardware when a low-speed SOF indication (keep-alive strobe) is seen on the USB bus. It is also issued at the end of a resume sequence.
0: No SOF signal detected
1: SOF signal detected

Note: To avoid spurious clearing of some bits, it is recommended to clear them using a load instruction where all bits which must not be altered are set, and all bits to be cleared are reset. Avoid read-modify-write instructions like AND, XOR..

INTERRUPT MASK REGISTER (IMR)

Read / Write

Reset Value: 0000 0000 (00h)

7							0
SUSPM	DOVRM	CTRM	ERRM	IOVRM	ESUSPM	RESETM	SOFM

Bits 7:0 = These bits are mask bits for all interrupt condition bits included in the ISTR. Whenever one of the IMR bits is set, if the corresponding ISTR bit is set, and the I bit in the CC register is cleared, an interrupt request is generated. For an explanation of each bit, please refer to the corresponding bit description in ISTR.

CONTROL REGISTER (CTLR)

Read / Write

Reset Value: 0000 0110 (06h)

7							0
0	0	0	0	RESUME	PDWN	FSUSP	FRES

Bits 7:4 = Reserved. Forced by hardware to 0.

Bit 3 = **RESUME** *Resume*.
This bit is set by software to wake-up the Host when the ST7 is in suspend mode.
0: Resume signal not forced
1: Resume signal forced on the USB bus.

Software should clear this bit after the appropriate delay.

Bit 2 = **PDWN** *Power down*.
This bit is set by software to turn off the 3.3V on-chip voltage regulator that supplies the external pull-up resistor and the transceiver.
0: Voltage regulator on
1: Voltage regulator off

Note: After turning on the voltage regulator, software should allow at least 3 µs for stabilisation of the power supply before using the USB interface.

Bit 1 = **FSUSP** *Force suspend mode*.
This bit is set by software to enter Suspend mode. The ST7 should also be halted allowing at least 600 ns before issuing the HALT instruction.
0: Suspend mode inactive
1: Suspend mode active

When the hardware detects USB activity, it resets this bit (it can also be reset by software).

Bit 0 = **FRES** *Force reset.*
This bit is set by software to force a reset of the USB interface, just as if a RESET sequence came from the USB.
0: Reset not forced
1: USB interface reset forced.

The USB is held in RESET state until software clears this bit, at which point a "USB-RESET" interrupt will be generated if enabled.

USB INTERFACE (Cont'd)

DEVICE ADDRESS REGISTER (DADDR)

Read / Write

Reset Value: 0000 0000 (00h)

7							0
0	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0

Bit 7 = Reserved. Forced by hardware to 0.

Bits 6:0 = **ADD[6:0]** *Device address, 7 bits.*

Software must write into this register the address sent by the host during enumeration.

Note: This register is also reset when a USB reset is received from the USB bus or forced through bit FRES in the CTLR register.

ENDPOINT n REGISTER A (EPnRA)

Read / Write

Reset Value: 0000 xxxx (0xh)

7							0
ST_ OUT	DTOG _TX	STAT _TX1	STAT _TX0	TBC 3	TBC 2	TBC 1	TBC 0

These registers (**EP0RA**, **EP1RA** and **EP2RA**) are used for controlling data transmission. They are also reset by the USB bus reset.

Note: Endpoint 2 and the EP2RA register are not available on some devices (see device feature list and register map).

Bit 7 = **ST_OUT** *Status out.*
This bit is set by software to indicate that a status out packet is expected: in this case, all nonzero OUT data transfers on the endpoint are STALLed instead of being ACKed. When ST_OUT is reset, OUT transactions can have any number of bytes, as needed.

Bit 6 = **DTOG_TX** *Data Toggle, for transmission transfers.*
It contains the required value of the toggle bit (0=DATA0, 1=DATA1) for the next transmitted data packet. This bit is set by hardware at the reception of a SETUP PID. DTOG_TX toggles only when the transmitter has received the ACK signal from the USB host. DTOG_TX and also DTOG_RX (see EPnRB) are normally updated by hardware, at the receipt of a relevant PID. They can be also written by software.

Bits 5:4 = **STAT_TX[1:0]** *Status bits, for transmission transfers.*
These bits contain the information about the endpoint status, which are listed below:

STAT_TX1	STAT_TX0	Meaning
0	0	**DISABLED:** transmission transfers cannot be executed.
0	1	**STALL**: the endpoint is stalled and all transmission requests result in a STALL handshake.
1	0	**NAK**: the endpoint is naked and all transmission requests result in a NAK handshake.
1	1	**VALID**: this endpoint is enabled for transmission.

These bits are written by software. Hardware sets the STAT_TX bits to NAK when a correct transfer has occurred (CTR=1) related to a IN or SETUP transaction addressed to this endpoint; this allows the software to prepare the next set of data to be transmitted.

Bits 3:0 = **TBC[3:0]** *Transmit byte count for Endpoint n.*
Before transmission, after filling the transmit buffer, software must write in the TBC field the transmit packet size expressed in bytes (in the range 0-8).

Warning: Any value outside the range 0-8 will induce undesired effects (such as continuous data transmission).

USB INTERFACE (Cont'd)

ENDPOINT n REGISTER B (EPnRB)

Read / Write

Reset Value: 0000 xxxx (0xh)

7							0
CTRL	DTOG _RX	STAT _RX1	STAT _RX0	EA3	EA2	EA1	EA0

These registers (**EP1RB** and **EP2RB**) are used for controlling data reception on Endpoints 1 and 2. They are also reset by the USB bus reset.

Note: Endpoint 2 and the EP2RB register are not available on some devices (see device feature list and register map).

Bit 7 = **CTRL** *Control.*
This bit should be 0.

Note: If this bit is 1, the Endpoint is a control endpoint. (Endpoint 0 is always a control Endpoint, but it is possible to have more than one control Endpoint).

Bit 6 = **DTOG_RX** *Data toggle, for reception transfers.*
It contains the expected value of the toggle bit (0=DATA0, 1=DATA1) for the next data packet. This bit is cleared by hardware in the first stage (Setup Stage) of a control transfer (SETUP transactions start always with DATA0 PID). The receiver toggles DTOG_RX only if it receives a correct data packet and the packet's data PID matches the receiver sequence bit.

Bits 5:4 = **STAT_RX [1:0]** *Status bits, for reception transfers.*
These bits contain the information about the endpoint status, which are listed below:

STAT_RX1	STAT_RX0	Meaning
0	0	**DISABLED**: reception transfers cannot be executed.
0	1	**STALL:** the endpoint is stalled and all reception requests result in a STALL handshake.
1	0	**NAK**: the endpoint is naked and all reception requests result in a NAK handshake.
1	1	**VALID**: this endpoint is enabled for reception.

These bits are written by software. Hardware sets the STAT_RX bits to NAK when a correct transfer has occurred (CTR=1) related to an OUT or SETUP transaction addressed to this endpoint, so the software has the time to elaborate the received data before acknowledging a new transaction.

Bits 3:0 = **EA[3:0]** *Endpoint address.*
Software must write in this field the 4-bit address used to identify the transactions directed to this endpoint. Usually EP1RB contains "0001" and EP2RB contains "0010".

ENDPOINT 0 REGISTER B (EP0RB)

Read / Write

Reset Value: 1000 0000 (80h)

7							0
1	DTOG RX	STAT RX1	STAT RX0	0	0	0	0

This register is used for controlling data reception on Endpoint 0. It is also reset by the USB bus reset.

Bit 7 = Forced by hardware to 1.

Bits 6:4 = Refer to the EPnRB register for a description of these bits.

Bits 3:0 = Forced by hardware to 0.

10.6.5 Programming Considerations

The interaction between the USB interface and the application program is described below. Apart from system reset, action is always initiated by the USB interface, driven by one of the USB events associated with the Interrupt Status Register (ISTR) bits.

10.6.5.1 Initializing the Registers

At system reset, the software must initialize all registers to enable the USB interface to properly generate interrupts and DMA requests.

1. Initialize the DMAR, IDR, and IMR registers (choice of enabled interrupts, address of DMA buffers). Refer the paragraph titled initializing the DMA Buffers.
2. Initialize the EP0RA and EP0RB registers to enable accesses to address 0 and endpoint 0 to support USB enumeration. Refer to the paragraph titled Endpoint Initialization.
3. When addresses are received through this channel, update the content of the DADDR.
4. If needed, write the endpoint numbers in the EA fields in the EP1RB and EP2RB register.

10.6.5.2 Initializing DMA buffers

The DMA buffers are a contiguous zone of memory whose maximum size is 48 bytes. They can be placed anywhere in the memory space to enable the reception of messages. The 10 most significant bits of the start of this memory area are specified by bits DA15-DA6 in registers DMAR and IDR, the remaining bits are 0. The memory map is shown in Figure 51.

Each buffer is filled starting from the bottom (last 3 address bits=000) up.

10.6.5.3 Endpoint Initialization

To be ready to receive:

Set STAT_RX to VALID (11b) in EP0RB to enable reception.

To be ready to transmit:

1. Write the data in the DMA transmit buffer.
2. In register EPnRA, specify the number of bytes to be transmitted in the TBC field
3. Enable the endpoint by setting the STAT_TX bits to VALID (11b) in EPnRA.

Note: Once transmission and/or reception are enabled, registers EPnRA and/or EPnRB (respectively) must not be modified by software, as the hardware can change their value on the fly.

When the operation is completed, they can be accessed again to enable a new operation.

10.6.5.4 Interrupt Handling

Start of Frame (SOF)

The interrupt service routine may monitor the SOF events for a 1 ms synchronization event to the USB bus. This interrupt is generated at the end of a resume sequence and can also be used to detect this event.

USB Reset (RESET)

When this event occurs, the DADDR register is reset, and communication is disabled in all endpoint registers (the USB interface will not respond to any packet). Software is responsible for reenabling endpoint 0 within 10 ms of the end of reset. To do this, set the STAT_RX bits in the EP0RB register to VALID.

Suspend (SUSP)

The CPU is warned about the lack of bus activity for more than 3 ms, which is a suspend request. The software should set the USB interface to suspend mode and execute an ST7 HALT instruction to meet the USB-specified power constraints.

End Suspend (ESUSP)

The CPU is alerted by activity on the USB, which causes an ESUSP interrupt. The ST7 automatically terminates HALT mode.

Correct Transfer (CTR)

1. When this event occurs, the hardware automatically sets the STAT_TX or STAT_RX to NAK.
   Note: Every valid endpoint is NAKed until software clears the CTR bit in the ISTR register, independently of the endpoint number addressed by the transfer which generated the CTR interrupt.
   Note: If the event triggering the CTR interrupt is a SETUP transaction, both STAT_TX and STAT_RX are set to NAK.
2. Read the PIDR to obtain the token and the IDR to get the endpoint number related to the last transfer.
   Note: When a CTR interrupt occurs, the TP3-TP2 bits in the PIDR register and EP1-EP0 bits in the IDR register stay unchanged until the CTR bit in the ISTR register is cleared.
3. Clear the CTR bit in the ISTR register.

Table 23. USB Register Map and Reset Values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
25	PIDR	TP3	TP2	0	0	0	RX_SEZ	RXD	0
	Reset Value	x	x	0	0	0	0	0	0
26	DMAR	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
	Reset Value	x	x	x	x	x	x	x	x
27	IDR	DA7	DA6	EP1	EP0	CNT3	CNT2	CNT1	CNT0
	Reset Value	x	x	x	x	0	0	0	0
28	ISTR	SUSP	DOVR	CTR	ERR	IOVR	ESUSP	RESET	SOF
	Reset Value	0	0	0	0	0	0	0	0
29	IMR	SUSPM	DOVRM	CTRM	ERRM	IOVRM	ESUSPM	RESETM	SOFM
	Reset Value	0	0	0	0	0	0	0	0
2A	CTLR	0	0	0	0	RESUME	PDWN	FSUSP	FRES
	Reset Value	0	0	0	0	0	1	1	0
2B	DADDR	0	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0
	Reset Value	0	0	0	0	0	0	0	0
2C	EP0RA	ST_OUT	DTOG_TX	STAT_TX1	STAT_TX0	TBC3	TBC2	TBC1	TBC0
	Reset Value	0	0	0	0	x	x	x	x
2D	EP0RB	1	DTOG_RX	STAT_RX1	STAT_RX0	0	0	0	0
	Reset Value	1	0	0	0	0	0	0	0
2E	EP1RA	ST_OUT	DTOG_TX	STAT_TX1	STAT_TX0	TBC3	TBC2	TBC1	TBC0
	Reset Value	0	0	0	0	x	x	x	x
2F	EP1RB	CTRL	DTOG_RX	STAT_RX1	STAT_RX0	EA3	EA2	EA1	EA0
	Reset Value	0	0	0	0	x	x	x	x
30	EP2RA	ST_OUT	DTOG_TX	STAT_TX1	STAT_TX0	TBC3	TBC2	TBC1	TBC0
	Reset Value	0	0	0	0	x	x	x	x
31	EP2RB	CTRL	DTOG_RX	STAT_RX1	STAT_RX0	EA3	EA2	EA1	EA0
	Reset Value	0	0	0	0	x	x	x	x

10.7 10-BIT A/D CONVERTER (ADC)

10.7.1 Introduction

The on-chip Analog to Digital Converter (ADC) peripheral is a 10-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has up to 8 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to 8 different sources.

The result of the conversion is stored in a 10-bit Data Register. The A/D converter is controlled through a Control/Status Register.

10.7.2 Main Features

- 10-bit conversion
- Up to 8 channels with multiplexed input
- Linear successive approximation
- Data register (DR) which contains the results
- Conversion complete status flag
- Continuous or One-Shot mode
- On/off bit (to reduce consumption)

The block diagram is shown in Figure 52.

10.7.3 Functional Description

10.7.3.1 Analog Power Supply

Depending on the MCU pin count, the package may feature separate V_{DDA} and V_{SSA} analog power supply pins. These pins supply power to the A/D converter cell and function as the high and low reference voltages for the conversion. In smaller packages V_{DDA} and V_{SSA} pins are not available and the analog supply and reference pads are internally bonded to the V_{DD} and V_{SS} pins.

Separation of the digital and analog power pins allow board designers to improve A/D performance. Conversion accuracy can be impacted by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines.

10.7.3.2 PCB Design Guidelines

To obtain best results, some general design and layout rules should be followed when designing the application PCB to shield the noise-sensitive, analog physical interface from noise-generating CMOS logic signals.

– Use separate digital and analog planes. The analog ground plane should be connected to the digital ground plane via a single point on the PCB. The analog power plane should be connected to the digital power plane via an RC network.

– Filter power to the analog power planes. The best solution is to connect a 0.1µF capacitor, with good high frequency characteristics, between V_{DDA} and V_{SSA} and place it as close as possible to the V_{DDA} and V_{SSA} pins and connect the analog and digital power supplies in a star network. Do not use a resistor, as V_{DDA} is used as a reference voltage by the A/D converter and resistance would cause a voltage drop and a loss of accuracy.

– Properly place components and route the signal traces on the PCB to shield the analog inputs. Analog signals paths should run over the analog ground plane and be as short as possible. Isolate analog signal from digital signals that may switch while the analog inputs are being sampled by the A/D converter. Do not toggle digital outputs on the same I/O port as the A/D input being converted.

10.7.3.3 Digital A/D Conversion Result

The conversion is monotonic, meaning that the result never decreases if the analog input does not and never increases if the analog input does not.

If the input voltage (V_{AIN}) is greater than V_{DDA} (high-level voltage reference) then the conversion result is FFh in the ADCDRMSB register and 03h in the ADCDRLSB register (without overflow indication).

If the input voltage (V_{AIN}) is lower than V_{SSA} (low-level voltage reference) then the conversion result in the ADCDRMSB and ADCDRLSB registers is 00 00h.

The A/D converter is linear and the digital result of the conversion is stored in the ADCDRMSB and ADCDRLSB registers. The accuracy of the conversion is described in the Electrical Characteristics Section.

R_{AIN} is the maximum recommended impedance for an analog input signal. If the impedance is too high, this will result in a loss of accuracy due to leakage and sampling not being completed in the allotted time.

10.7.3.4 A/D Conversion

Conversion can be performed in One-Shot or Continuous mode. Continuous mode is typically used for monitoring a single channel. One-shot mode should be used when the application requires inputs from several channels.

ADC Configuration

The analog input ports must be configured as input, no pull-up, no interrupt. Refer to the «I/O ports» chapter. Using these pins as analog inputs does not affect the ability of the port to be read as a logic input.

In the ADCCSR register:

– Select the CS[2:0] bits to assign the analog channel to convert.

ADC One-Shot Conversion mode

In the ADCCSR register:

1.Set the ONE SHOT bit to put the A/D converter in one shot mode.

2.Set the ADON bit to enable the A/D converter and to start the conversion. The EOC bit is kept low by hardware during the conversion.

Note: Changing the A/D channel during conversion will stop the current conversion and start conversion of the newly selected channel.

When a conversion is complete:

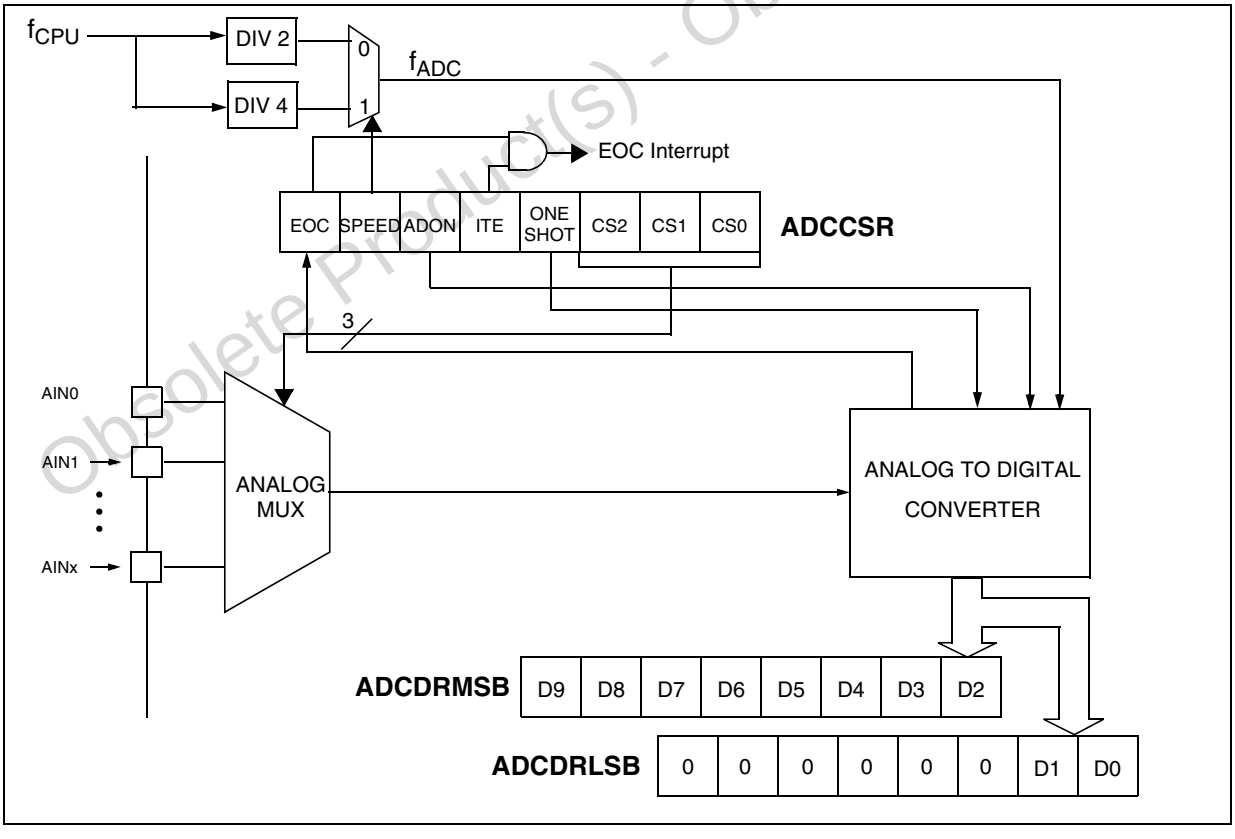
– The EOC bit is set by hardware.
– An interrupt request is generated if the ITE bit is set.
– The ADON bit is reset by hardware.
– The result is in the ADCDR registers.

To read the 10 bits, perform the following steps:

1. Wait for interrupt or poll the EOC bit
2. Read ADCDRLSB
3. Read ADCDRMSB

The EOC bit is reset by hardware once the ADCDRMSB is read.

Figure 52. ADC Block Diagram



To read only 8 bits, perform the following steps:

1. Wait for interrupt or poll the EOC bit
2. Read ADCDRMSB

The EOC bit is reset by hardware once the ADCDRMSB is read.

To start another conversion, user should set the ADON bit once again.

ADC Continuous Conversion mode

In the ADCCSR register:

1.Reset the ONE SHOT bit to put the A/D converter in continuous mode.

2.Set the ADON bit to enable the A/D converter and to start the first conversion. From this time on, the ADC performs a continuous conversion of the selected channel.

Note: Changing the A/D channel during conversion will stop the current conversion and start conversion of the newly selected channel.

When a conversion is complete:

- The EOC bit is set by hardware.
- An interrupt request is generated if the ITE bit is set.
- The result is in the ADCDR registers and remains valid until the next conversion has ended.

To read the 10 bits, perform the following steps:

1. Wait for interrupt or poll the EOC bit
2. Read ADCDRLSB
3. Read ADCDRMSB

The EOC bit is reset by hardware once the ADCDRMSB is read.

To read only 8 bits, perform the following steps:

1. Wait for interrupt
2. Read ADCDRMSB

The EOC bit is reset by hardware once the ADCDRMSB is read.

Changing the conversion channel

The application can change channels during conversion. In this case the current conversion is stopped and the A/D converter starts converting the newly selected channel.

ADCCR consistency

If an End Of Conversion event occurs after software has read the ADCDRLSB but before it has read the ADCDRMSB, there would be a risk that the two values read would belong to different samples.

To guarantee consistency:

- The ADCDRMSB and the ADCDRLSB are locked when the ADCCRLSB is read
- The ADCDRMSB and the ADCDRLSB are unlocked when the MSB is read or when ADON is reset.

Thus, it is mandatory to read the ADCDRMSB just after reading the ADCDRLSB. This is especially important in continuous mode, as the ADCDR register will not be updated until the ADCDRMSB is read.

10.7.4 Low Power Modes

Note: The A/D converter may be disabled by resetting the ADON bit. This feature allows reduced power consumption when no conversion is needed and between single shot conversions.

Mode	Description
WAIT	No effect on A/D Converter
HALT	A/D Converter disabled. After wakeup from Halt mode, the A/D Converter requires a stabilisation time t_{STAB} (see Electrical Characteristics) before accurate conversions can be performed.

10.7.5 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
End of Conversion	EOC	ITE	Yes	No

Note: The EOC interrupt event is connected to an interrupt vector (see Interrupts chapter).
It generates an interrupt if the ITE bit is set in the ADCCSR register and the interrupt mask in the CC register is reset (RIM instruction).

10.7.6 Register Description

CONTROL/STATUS REGISTER (ADCCSR)

Read/Write (Except bit 7 read only)

Reset Value: 0000 0000 (00h)

7							0
EOC	SPEED	ADON	ITE	ONE SHOT	CS2	CS1	CS0

Bit 7 = **EOC** *End of Conversion*
This bit is set by hardware. It is cleared by software reading the ADCDRMSB register.
0: Conversion is not complete
1: Conversion complete

Bit 6 = **SPEED** *ADC clock selection*
This bit is set and cleared by software.
0: $f_{ADC} = f_{CPU}/2$
1: $f_{ADC} = f_{CPU}/4$

Bit 5 = **ADON** *A/D Converter on*
This bit is set and cleared by software or by hardware after the end of a one shot conversion.
0: Disable ADC and stop conversion
1: Enable ADC and start conversion

Bit 4 = **ITE** *Interrupt Enable*
This bit is set and cleared by software.
0: EOC Interrupt disabled
1: EOC Interrupt enabled

Bit 3 = **ONESHOT** *One Shot Conversion Selection*
This bit is set and cleared by software.
0: Continuous conversion mode
1: One Shot conversion mode

Bit 2:0 = **CS[2:0]** *Channel Selection*
These bits are set and cleared by software. They select the analog input to convert.

Channel*	CS2	CS1	CS0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

*The number of channels is device dependent. Refer to the device pinout description.

DATA REGISTER (ADCDRMSB)

Read Only

Reset Value: 0000 0000 (00h)

7							0
D9	D8	D7	D6	D5	D4	D3	D2

Bit 7:0 = **D[9:2]** *MSB of Analog Converted Value*
This register contains the MSB of the converted analog value.

DATA REGISTER (ADCDRLSB)

Read Only

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	D1	D0

Bit 7:2 = Reserved. Forced by hardware to 0.

Bit 1:0 = **D[1:0]** *LSB of Analog Converted Value*
This register contains the LSB of the converted analog value.

Note: please refer to Section 15 IMPORTANT NOTES

11 INSTRUCTION SET

11.1 CPU ADDRESSING MODES

The CPU features 17 different addressing modes which can be classified in seven main groups:

Addressing Mode	Example
Inherent	nop
Immediate	ld A,#$55
Direct	ld A,$55
Indexed	ld A,($55,X)
Indirect	ld A,([$55],X)
Relative	jrne loop
Bit operation	bset byte,#5

The CPU Instruction set is designed to minimize the number of bytes required per instruction: To do so, most of the addressing modes may be subdivided in two submodes called long and short:

– Long addressing mode is more powerful because it can use the full 64 Kbyte address space, however it uses more bytes and more CPU cycles.

– Short addressing mode is less powerful because it can generally only access page zero (0000h - 00FFh range), but the instruction size is more compact, and faster. All memory to memory instructions use short addressing modes only (CLR, CPL, NEG, BSET, BRES, BTJT, BTJF, INC, DEC, RLC, RRC, SLL, SRL, SRA, SWAP)

The ST7 Assembler optimizes the use of long and short addressing modes.

Table 24. CPU Addressing Mode Overview

Mode			Syntax	Destination	Pointer Address (Hex.)	Pointer Size (Hex.)	Length (Bytes)
Inherent			nop				+ 0
Immediate			ld A,#$55				+ 1
Short	Direct		ld A,$10	00..FF			+ 1
Long	Direct		ld A,$1000	0000..FFFF			+ 2
No Offset	Direct	Indexed	ld A,(X)	00..FF			+ 0
Short	Direct	Indexed	ld A,($10,X)	00..1FE			+ 1
Long	Direct	Indexed	ld A,($1000,X)	0000..FFFF			+ 2
Short	Indirect		ld A,[$10]	00..FF	00..FF	byte	+ 2
Long	Indirect		ld A,[$10.w]	0000..FFFF	00..FF	word	+ 2
Short	Indirect	Indexed	ld A,([$10],X)	00..1FE	00..FF	byte	+ 2
Long	Indirect	Indexed	ld A,([$10.w],X)	0000..FFFF	00..FF	word	+ 2
Relative	Direct		jrne loop	PC+/-127			+ 1
Relative	Indirect		jrne [$10]	PC+/-127	00..FF	byte	+ 2
Bit	Direct		bset $10,#7	00..FF			+ 1
Bit	Indirect		bset [$10],#7	00..FF	00..FF	byte	+ 2
Bit	Direct	Relative	btjt $10,#7,skip	00..FF			+ 2
Bit	Indirect	Relative	btjt [$10],#7,skip	00..FF	00..FF	byte	+ 3

11.1.1 Inherent

All Inherent instructions consist of a single byte. The opcode fully specifies all the required information for the CPU to process the operation.

Inherent Instruction	Function
NOP	No operation
TRAP	S/W Interrupt
WFI	Wait For Interrupt (Low Power Mode)
HALT	Halt Oscillator (Lowest Power Mode)
RET	Sub-routine Return
IRET	Interrupt Sub-routine Return
SIM	Set Interrupt Mask (level 3)
RIM	Reset Interrupt Mask (level 0)
SCF	Set Carry Flag
RCF	Reset Carry Flag
RSP	Reset Stack Pointer
LD	Load
CLR	Clear
PUSH/POP	Push/Pop to/from the stack
INC/DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
MUL	Byte Multiplication
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles

11.1.2 Immediate

Immediate instructions have 2 bytes, the first byte contains the opcode, the second byte contains the operand value.

Immediate Instruction	Function
LD	Load
CP	Compare
BCP	Bit Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Operations

11.1.3 Direct

In Direct instructions, the operands are referenced by their memory address.

The direct addressing mode consists of two submodes:

Direct (short)

The address is a byte, thus requires only one byte after the opcode, but only allows 00 - FF addressing space.

Direct (long)

The address is a word, thus allowing 64 Kbyte addressing space, but requires 2 bytes after the opcode.

11.1.4 Indexed (No Offset, Short, Long)

In this mode, the operand is referenced by its memory address, which is defined by the unsigned addition of an index register (X or Y) with an offset.

The indirect addressing mode consists of three submodes:

Indexed (No Offset)

There is no offset, (no extra byte after the opcode), and allows 00 - FF addressing space.

Indexed (Short)

The offset is a byte, thus requires only one byte after the opcode and allows 00 - 1FE addressing space.

Indexed (long)

The offset is a word, thus allowing 64 Kbyte addressing space and requires 2 bytes after the opcode.

11.1.5 Indirect (Short, Long)

The required data byte to do the operation is found by its memory address, located in memory (pointer).

The pointer address follows the opcode. The indirect addressing mode consists of two submodes:

Indirect (short)

The pointer address is a byte, the pointer size is a byte, thus allowing 00 - FF addressing space, and requires 1 byte after the opcode.

Indirect (long)

The pointer address is a byte, the pointer size is a word, thus allowing 64 Kbyte addressing space, and requires 1 byte after the opcode.

11.1.6 Indirect Indexed (Short, Long)

This is a combination of indirect and short indexed addressing modes. The operand is referenced by its memory address, which is defined by the unsigned addition of an index register value (X or Y) with a pointer value located in memory. The pointer address follows the opcode.

The indirect indexed addressing mode consists of two submodes:

Indirect Indexed (Short)

The pointer address is a byte, the pointer size is a byte, thus allowing 00 - 1FE addressing space, and requires 1 byte after the opcode.

Indirect Indexed (Long)

The pointer address is a byte, the pointer size is a word, thus allowing 64 Kbyte addressing space, and requires 1 byte after the opcode.

Table 25. Instructions Supporting Direct, Indexed, Indirect and Indirect Indexed Addressing Modes

Long and Short Instructions	Function
LD	Load
CP	Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Additions/Substractions operations
BCP	Bit Compare

Short Instructions Only	Function
CLR	Clear
INC, DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
BSET, BRES	Bit Operations
BTJT, BTJF	Bit Test and Jump Operations
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles
CALL, JP	Call or Jump subroutine

11.1.7 Relative mode (Direct, Indirect)

This addressing mode is used to modify the PC register value, by adding an 8-bit signed offset to it.

Available Relative Direct/Indirect Instructions	Function
JRxx	Conditional Jump
CALLR	Call Relative

The relative addressing mode consists of two submodes:

Relative (Direct)

The offset is following the opcode.

Relative (Indirect)

The offset is defined in memory, which address follows the opcode.

11.2 INSTRUCTION GROUPS

The ST7 family devices use an Instruction Set consisting of 63 instructions. The instructions may be subdivided into 13 main groups as illustrated in the following table:

Load and Transfer	LD	CLR						
Stack operation	PUSH	POP	RSP					
Increment/Decrement	INC	DEC						
Compare and Tests	CP	TNZ	BCP					
Logical operations	AND	OR	XOR	CPL	NEG			
Bit Operation	BSET	BRES						
Conditional Bit Test and Branch	BTJT	BTJF						
Arithmetic operations	ADC	ADD	SUB	SBC	MUL			
Shift and Rotates	SLL	SRL	SRA	RLC	RRC	SWAP	SLA	
Unconditional Jump or Call	JRA	JRT	JRF	JP	CALL	CALLR	NOP	RET
Conditional Branch	JRxx							
Interruption management	TRAP	WFI	HALT	IRET				
Condition Code Flag modification	SIM	RIM	SCF	RCF				

Using a prebyte

The instructions are described with one to four opcodes.

In order to extend the number of available opcodes for an 8-bit CPU (256 opcodes), three different prebyte opcodes are defined. These prebytes modify the meaning of the instruction they precede.

The whole instruction becomes:

PC-2 End of previous instruction

PC-1 Prebyte

PC Opcode

PC+1 Additional word (0 to 2) according to the number of bytes required to compute the effective address

These prebytes enable instruction in Y as well as indirect addressing modes to be implemented. They precede the opcode of the instruction in X or the instruction using direct addressing mode. The prebytes are:

PDY 90 Replace an X based instruction using immediate, direct, indexed, or inherent addressing mode by a Y one.

PIX 92 Replace an instruction using direct, direct bit, or direct relative addressing mode to an instruction using the corresponding indirect addressing mode.
It also changes an instruction using X indexed addressing mode to an instruction using indirect X indexed addressing mode.

PIY 91 Replace an instruction using X indirect indexed addressing mode by a Y one.

INSTRUCTION SET OVERVIEW (Cont'd)

Mnemo	Description	Function/Example	Dst	Src		I1	H	I0	N	Z	C
ADC	Add with Carry	A = A + M + C	A	M			H		N	Z	C
ADD	Addition	A = A + M	A	M			H		N	Z	C
AND	Logical And	A = A . M	A	M					N	Z	
BCP	Bit compare A, Memory	tst (A . M)	A	M					N	Z	
BRES	Bit Reset	bres Byte, #3	M								
BSET	Bit Set	bset Byte, #3	M								
BTJF	Jump if bit is false (0)	btjf Byte, #3, Jmp1	M								C
BTJT	Jump if bit is true (1)	btjt Byte, #3, Jmp1	M								C
CALL	Call subroutine										
CALLR	Call subroutine relative										
CLR	Clear		reg, M						0	1	
CP	Arithmetic Compare	tst(Reg - M)	reg	M					N	Z	C
CPL	One Complement	A = FFH-A	reg, M						N	Z	1
DEC	Decrement	dec Y	reg, M						N	Z	
HALT	Halt					1		0			
IRET	Interrupt routine return	Pop CC, A, X, PC				I1	H	I0	N	Z	C
INC	Increment	inc X	reg, M						N	Z	
JP	Absolute Jump	jp [TBL.w]									
JRA	Jump relative always										
JRT	Jump relative										
JRF	Never jump	jrf *									
JRIH	Jump if ext. INT pin = 1	(ext. INT pin high)									
JRIL	Jump if ext. INT pin = 0	(ext. INT pin low)									
JRH	Jump if H = 1	H = 1 ?									
JRNH	Jump if H = 0	H = 0 ?									
JRM	Jump if I1:0 = 11	I1:0 = 11 ?									
JRNM	Jump if I1:0 <> 11	I1:0 <> 11 ?									
JRMI	Jump if N = 1 (minus)	N = 1 ?									
JRPL	Jump if N = 0 (plus)	N = 0 ?									
JREQ	Jump if Z = 1 (equal)	Z = 1 ?									
JRNE	Jump if Z = 0 (not equal)	Z = 0 ?									
JRC	Jump if C = 1	C = 1 ?									
JRNC	Jump if C = 0	C = 0 ?									
JRULT	Jump if C = 1	Unsigned <									
JRUGE	Jump if C = 0	Jmp if unsigned >=									
JRUGT	Jump if (C + Z = 0)	Unsigned >									

INSTRUCTION SET OVERVIEW (Cont'd)

Mnemo	Description	Function/Example	Dst	Src		I1	H	I0	N	Z	C
JRULE	Jump if (C + Z = 1)	Unsigned <=									
LD	Load	dst <= src	reg, M	M, reg					N	Z	
MUL	Multiply	X,A = X * A	A, X, Y	X, Y, A			0				0
NEG	Negate (2's compl)	neg $10	reg, M						N	Z	C
NOP	No Operation										
OR	OR operation	A = A + M	A	M					N	Z	
POP	Pop from the Stack	pop reg	reg	M							
		pop CC	CC	M		I1	H	I0	N	Z	C
PUSH	Push onto the Stack	push Y	M	reg, CC							
RCF	Reset carry flag	C = 0									0
RET	Subroutine Return										
RIM	Enable Interrupts	I1:0 = 10 (level 0)				1		0			
RLC	Rotate left true C	C <= A <= C	reg, M						N	Z	C
RRC	Rotate right true C	C => A => C	reg, M						N	Z	C
RSP	Reset Stack Pointer	S = Max allowed									
SBC	Substract with Carry	A = A - M - C	A	M					N	Z	C
SCF	Set carry flag	C = 1									1
SIM	Disable Interrupts	I1:0 = 11 (level 3)				1		1			
SLA	Shift left Arithmetic	C <= A <= 0	reg, M						N	Z	C
SLL	Shift left Logic	C <= A <= 0	reg, M						N	Z	C
SRL	Shift right Logic	0 => A => C	reg, M						0	Z	C
SRA	Shift right Arithmetic	A7 => A => C	reg, M						N	Z	C
SUB	Substraction	A = A - M	A	M					N	Z	C
SWAP	SWAP nibbles	A7-A4 <=> A3-A0	reg, M						N	Z	
TNZ	Test for Neg & Zero	tnz lbl1							N	Z	
TRAP	S/W trap	S/W interrupt				1		1			
WFI	Wait for Interrupt					1		0			
XOR	Exclusive OR	A = A XOR M	A	M					N	Z	

12 ELECTRICAL CHARACTERISTICS

12.1 PARAMETER CONDITIONS

Unless otherwise specified, all voltages are referred to V_{SS}.

12.1.1 Minimum and Maximum Values

Unless otherwise specified the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage and frequencies by tests in production on 100% of the devices with an ambient temperature at T_A=25°C and T_A=$T_A$max (given by the selected temperature range).

Data based on characterization results, design simulation and/or technology characteristics are indicated in the table footnotes and are not tested in production. Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value plus or minus three times the standard deviation (mean±3Σ).

12.1.2 Typical Values

Unless otherwise specified, typical data are based on T_A=25°C, V_{DD}=5V. They are given only as design guidelines and are not tested.

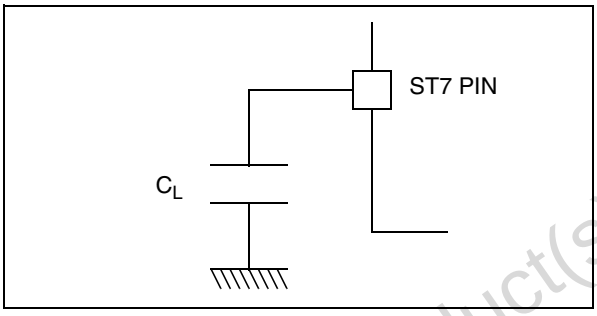
12.1.3 Typical Curves

Unless otherwise specified, all typical curves are given only as design guidelines and are not tested.

12.1.4 Loading Capacitor

The loading conditions used for pin parameter measurement are shown in Figure 53.

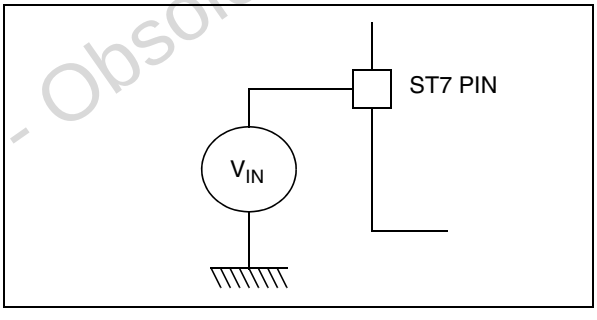
Figure 53. Pin Loading Conditions



12.1.5 Pin Input Voltage

The input voltage measurement on a pin of the device is described in Figure 54.

Figure 54. Pin Input Voltage



12.2 ABSOLUTE MAXIMUM RATINGS

Stresses above those listed as “absolute maximum ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device under these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

12.2.1 Voltage Characteristics

Symbol	Ratings	Maximum value	Unit
V_{DD} - V_{SS}	Supply voltage	6.0	V
V_{DDA} - V_{SSA}	Analog Reference Voltage	6.0	
V_{IN} 1) & 2)	Input voltage on true open drain pin	V_{SS}-0.3 to 6.0	
	Input voltage on any other pin	V_{SS}-0.3 to V_{DD}+0.3	
$V_{ESD(HBM)}$	Electrostatic discharge voltage (Human Body Model)	See “Electrostatic Discharge (ESD)” on page 109.	

12.2.2 Current Characteristics

Symbol	Ratings	Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source) 3)	80	mA
I_{VSS}	Total current out of V_{SS} ground lines (sink) 3)	80	
I_{IO}	Output current sunk by any standard I/O and control pin	25	
	Output current sunk by any high sink I/O pin	50	
	Output current source by any I/Os and control pin	- 25	
$I_{INJ(PIN)}$ 2) & 4)	Injected current on V_{PP} pin	75	
	Injected current on $\overline{RESET}$ pin	± 5	
	Injected current on OSCIN and OSCOUT pins	± 5	
	Injected current on PA0 to PA6 pins	± 5	
	Injected current on PA7 pin	+ 5	
	Injected current on any other pin 5) & 6)	± 5	
$\Sigma I_{INJ(PIN)}$ 2)	Total injected current (sum of all I/O and control pins) 5)	± 20	

Notes:

1. Directly connecting the $\overline{RESET}$ and I/O pins to V_{DD} or V_{SS} could damage the device if an unintentional internal reset is generated or an unexpected change of the I/O configuration occurs (for example, due to a corrupted program counter). To guarantee safe operation, this connection has to be done through a pull-up or pull-down resistor (typical: 4.7kΩ for $\overline{RESET}$, 10kΩ for I/Os). Unused I/O pins must be tied in the same way to V_{DD} or V_{SS} according to their reset configuration.

2. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by $V_{IN}>V_{DD}$ while a negative injection is induced by $V_{IN}<V_{SS}$. For true open-drain pads, there is no positive injection current, and the corresponding V_{IN} maximum must always be respected

3. All power (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.

4. Negative injection disturbs the analog performance of the device. In particular, it induces leakage currents throughout the device including the analog inputs. To avoid undesirable effects on the analog functions, care must be taken:
- Analog input pins must have a negative injection less than 0.8 mA (assuming that the impedance of the analog voltage is lower than the specified limits)
- Pure digital pins must have a negative injection less than 1.6mA. In addition, it is recommended to inject the current as far as possible from the analog input pins.

5. When several inputs are submitted to a current injection, the maximum $\Sigma I_{INJ(PIN)}$ is the absolute sum of the positive and negative injected currents (instantaneous values). These results are based on characterization with $\Sigma I_{INJ(PIN)}$ maximum current injection on four I/O port pins of the device.

6. True open drain I/O port pins do not accept positive injection.

12.2.3 Thermal Characteristics

Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	-65 to +150	°C
T_J	Maximum junction temperature[1)]	175	°C

Notes:
1. The maximum chip-junction temperature is based on technology characteristics.

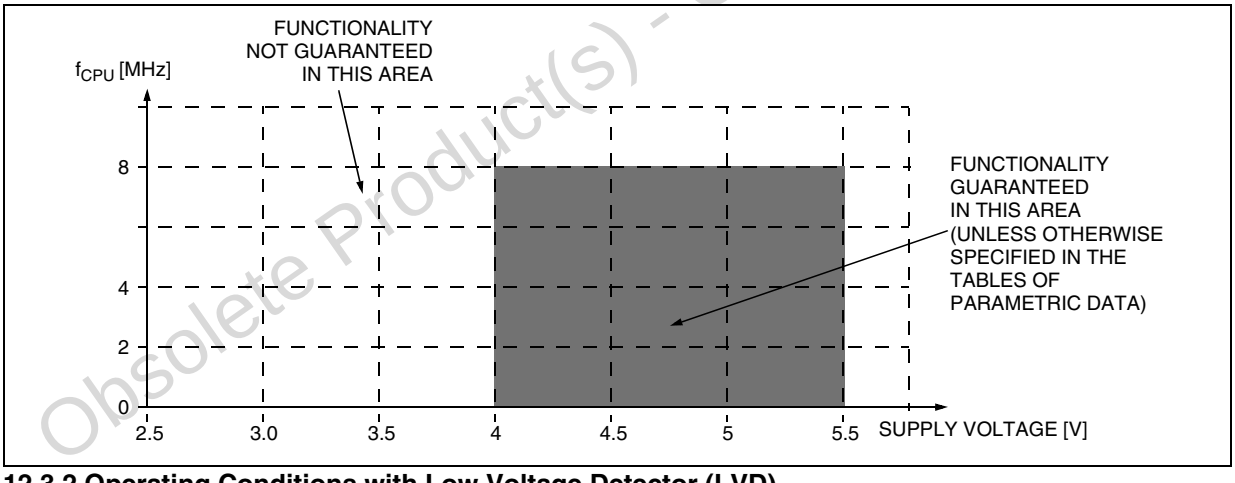
12.3 OPERATING CONDITIONS

12.3.1 General Operating Conditions (standard voltage ROM and Flash devices)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DD}	Operating Supply Voltage	f_{CPU} = 8 MHz	4	5	5.5	
V_{DDA}	Analog reference voltage		V_{DD}		V_{DD}	
V_{SSA}	Analog reference voltage		V_{SS}		V_{SS}	
f_{CPU}	Operating frequency	f_{OSC} = 12MHz			8	MHz
		f_{OSC} = 6MHz			4	
T_A	Ambient temperature range		0		70	°C

.

Figure 55. f_{CPU} Versus V_{DD} for standard voltage devices



12.3.2 Operating Conditions with Low Voltage Detector (LVD)

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A. Refer to Figure 15 on page 21.

Symbol	Parameter	Conditions	Min	Typ [1)]	Max	Unit
V_{IT+}	Low Voltage Reset Threshold (V_{DD} rising)	V_{DD} Max. Variation 50V/ms	3.6	3.8	3.95	V
V_{IT-}	Low Voltage Reset Threshold (V_{DD} falling)	V_{DD} Max. Variation 50V/ms	3.45	3.65	3.8	V
V_{hyst}	Hysteresis (V_{IT+} - V_{IT-})		120 [2)]	150 [2)]	180 [2)]	mV
Vt_{POR}	V_{DD} rise time rate [3)]		0.5		50	V/ms

Notes:
1. Not tested, guaranteed by design.
2. Not tested in production, guaranteed by characterization.
3. The V_{DD} rise time rate condition is needed to insure a correct device power-on and LVD reset. Not tested in production.

12.4 SUPPLY CURRENT CHARACTERISTICS

The following current consumption specified for the ST7 functional operating modes over temperature range does not take into account the clock source current consumption. To get the total device consumption, the two current values must be added (except for HALT mode for which the clock is stopped).

Symbol	Parameter	Conditions		Typ [1]	Max	Unit
$\Delta I_{DD(\Delta Ta)}$	Supply current variation vs. temperature	Constant V_{DD} and f_{CPU}			10	%
I_{DD}	CPU RUN mode	I/Os in input mode. USB transceiver and LVD disabled	f_{CPU} = 4 MHz	6	8	mA
			f_{CPU} = 8 MHz	8	14	
		LVD enabled. USB in Transmission[2]	f_{CPU} = 4 MHz	13	18	mA
			f_{CPU} = 8 MHz	15	24	mA
	CPU WAIT mode[2]	I/Os in input mode. USB transceiver and LVD disabled	f_{CPU} = 8 MHz	7	12	mA
		LVD enabled. USB in Transmission	f_{CPU} = 8 MHz	14	22	mA
	CPU HALT mode[3]	with LVD		130	200	µA
		without LVD		30	50	
	USB Suspend mode[4]			130	200	µA
$I_{DD(ADC)}$	ADC supply current when converting	f_{ADC}=4MHz		1000[2]		µA

Note 1: Typical data are based on T_A=25°C and not tested in production

Note 2: Data based on design simulation, not tested in production.

Note 3: USB Transceiver and ADC are powered down.

Note 4: Low voltage reset function enabled.
CPU in HALT mode.
Current consumption of external pull-up (1.5Kohms to USBVCC) and pull-down (15Kohms to V_{SSA}) not included.

Figure 56. Typ. I_{DD} in RUN at 4 and 8 MHz f_{CPU}

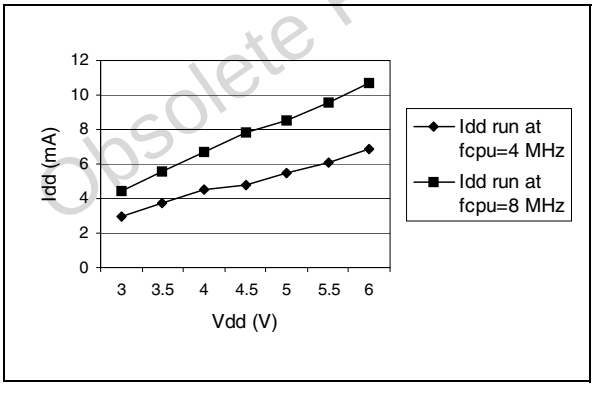
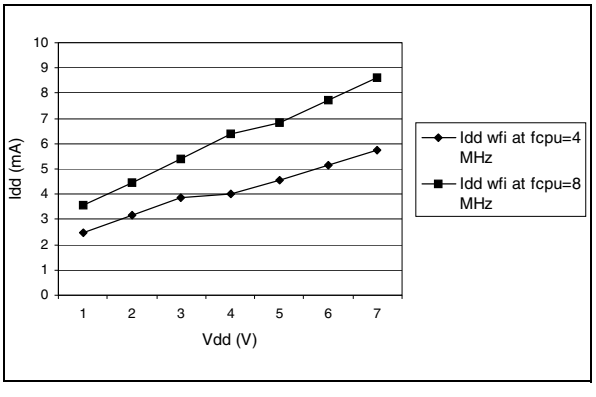


Figure 57. Typ. I_{DD} in WAIT at 4 and 8 MHz f_{CPU}



12.5 clOck and timing characteristics

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A.

12.5.1 General Timings

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
$t_{c(INST)}$	Instruction cycle time	f_{CPU}=8MHz	2	3	12	t_{CPU}
			250	375	1500	ns
$t_{v(IT)}$	Interrupt reaction time [2] $t_{v(IT)} = \Delta t_{c(INST)} + 10\ t_{CPU}$	f_{CPU}=8MHz	10		22	t_{CPU}
			1.25		2.75	µs

1. Data based on typical application software.
2. Time measured between interrupt event and interrupt vector fetch. $\Delta t_{c(INST)}$ is the number of t_{CPU} cycles needed to finish the current instruction execution.

12.5.2 Control timing characteristics

CONTROL TIMINGS						
Symbol	Parameter	Conditions	Value			Unit
			Min [2]	Typ. [2]	Max [2]	
f_{OSC}	Oscillator Frequency				12	MHz
f_{CPU}	Operating Frequency				8	MHz
t_{RL}	External RESET Input pulse Width		1.5			t_{CPU}
t_{PORL}	Internal Power Reset Duration		514			t_{CPU}
T_{RSTL}	Reset Pin Output Pulse Width		10			µs
t_{WDG}	Watchdog Time-out	f_{cpu} = 8MHz	65536 8.192		4194304 524.288	t_{CPU} ms
t_{OXOV}	Crystal Oscillator Start-up Time		20	30	40	ms
t_{DDR}	Power up rise time	from V_{DD} = 0 to 4V			100	ms

Note 1:
The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 cycles.

Note 2:
Not tested in production, guaranteed by design.

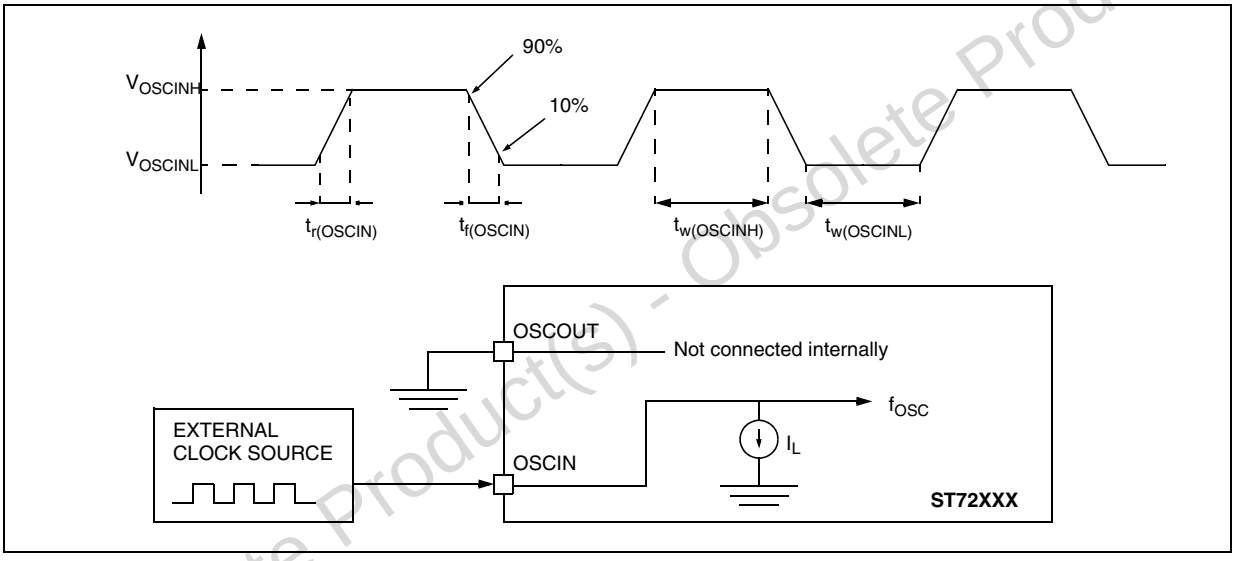
12.5.3 External Clock Source

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OSCINH}	OSCIN input pin high level voltage	see Figure 58	$0.7xV_{DD}$		V_{DD}	V
V_{OSCINL}	OSCIN input pin low level voltage		V_{SS}		$0.3xV_{DD}$	
$t_{w(OSCINH)}$ $t_{w(OSCINL)}$	OSCIN high or low time 1)		15			ns
$t_{r(OSCIN)}$ $t_{f(OSCIN)}$	OSCIN rise or fall time1)				15	
I_L	OSCx Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$			±1	µA

Note:

1. Refer to Figure 58 for more information.

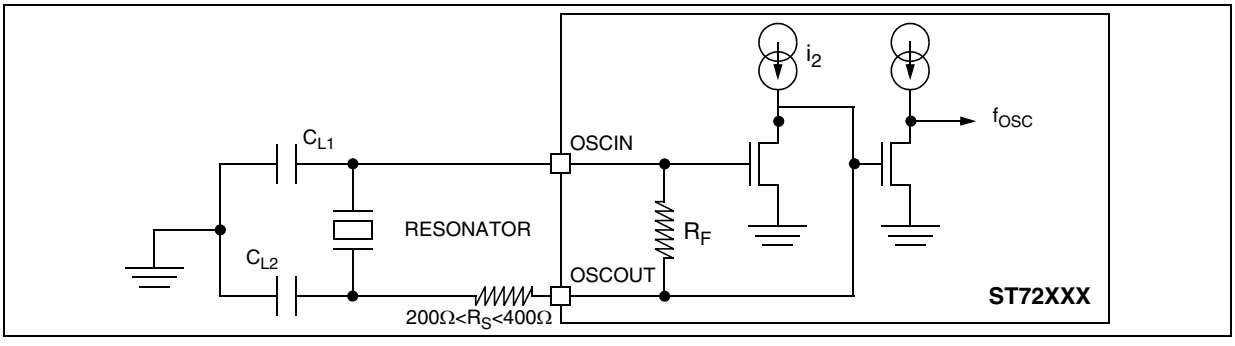
Figure 58. Typical Application with an External Clock Source



12.5.4 Crystal Oscillator Output Drive Level

Symbol	Parameter	Conditions	Typ	Unit
P_{OSCOUT}	Oscillator OSCOUT pin drive level	At 5V / 25°C	1	mW

Figure 59. Typical Application with a Crystal Resonator



12.6 MEMORY CHARACTERISTICS

Subject to general operating conditions for f_{CPU}, and T_A unless otherwise specified.

12.6.1 RAM and Hardware Registers

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{RM}	Data retention mode [1]	HALT mode (or RESET)	2.0			V

Note 1: Guaranteed by design. Not tested in production.

12.6.2 FLASH Memory

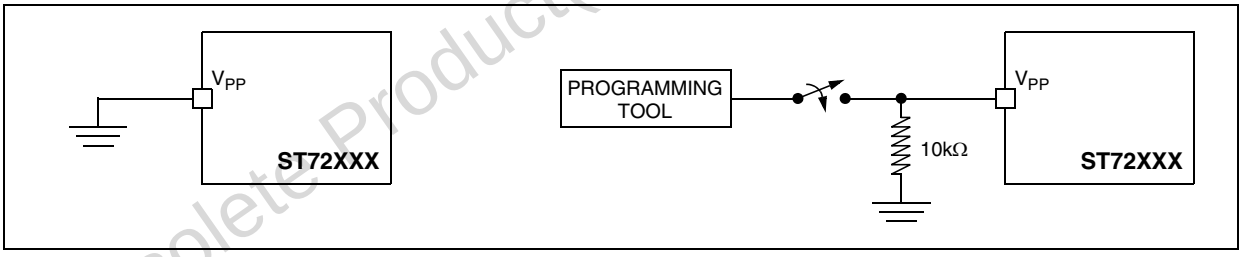
Operating Conditions: f_{CPU} = 8 MHz.

DUAL VOLTAGE FLASH MEMORY [1]						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
f_{CPU}	Operating Frequency	Read mode			8	MHz
		Write / Erase mode, T_A=25°C			8	
V_{PP}	Programming Voltage	4.0V <= V_{DD} <= 5.5V	11.4		12.6	V
I_{PP}	V_{PP} Current	Write / Erase			30	mA
t_{VPP}	Internal V_{PP} Stabilization Time			10		µs
t_{RET}	Data Retention	$T_A \leq 55°C$	40			years
N_{RW}	Write Erase Cycles	T_A=25°C	100			cycles

Note:

1. Refer to the Flash Programming Reference Manual for the typical HDFlash programming and erase timing values.

Figure 60. Two typical Applications with V_{PP} Pin [1]



Note 1: When the ICP mode is not required by the application, V_{PP} pin must be tied to V_{SS}.

12.7 EMC CHARACTERISTICS

Susceptibility tests are performed on a sample basis during product characterization.

12.7.1 Functional EMS (Electromagnetic Susceptibility)

Based on a simple running application on the product (toggling 2 LEDs through I/O ports), the product is stressed by two electromagnetic events until a failure occurs (indicated by the LEDs).

- **ESD**: Electrostatic Discharge (positive and negative) is applied on all pins of the device until a functional disturbance occurs. This test conforms with the IEC 1000-4-2 standard.
- **FTB**: A Burst of Fast Transient voltage (positive and negative) is applied to V_{DD} and V_{SS} through a 100pF capacitor, until a functional disturbance occurs. This test conforms with the IEC 1000-4-4 standard.

A device reset allows normal operations to be resumed. The test results are given in the table below based on the EMS levels and classes defined in application note AN1709.

12.7.1.1 Designing hardened software to avoid noise problems

EMC characterization and optimization are performed at component level with a typical application environment and simplified MCU software. It should be noted that good EMC performance is highly dependent on the user application and the software in particular.

Therefore it is recommended that the user applies EMC software optimization and prequalification tests in relation with the EMC level requested for his application.

Software recommendations:

The software flowchart must include the management of runaway conditions such as:

– Corrupted program counter

– Unexpected reset

– Critical Data corruption (control registers...)

Prequalification trials:

Most of the common failures (unexpected reset and program counter corruption) can be reproduced by manually forcing a low state on the RESET pin or the Oscillator pins for 1 second.

To complete these trials, ESD stress can be applied directly on the device, over the range of specification values. When unexpected behaviour is detected, the software can be hardened to prevent unrecoverable errors occurring (see application note AN1015).

Symbol	Parameter	Conditions	Level/ Class
V_{FESD}	Voltage limits to be applied on any I/O pin to induce a functional disturbance	V_{DD}=5V, T_A=+25°C, PDIP42, f_{CPU}=8MHz conforms to IEC 1000-4-2	2B
V_{FFTB}	Fast transient voltage burst limits to be applied through 100pF on V_{DD} and V_{SS} pins to induce a functional disturbance	V_{DD}=5V, T_A=+25°C, PDIP42, f_{CPU}=8MHz conforms to IEC 1000-4-4	2B

12.7.2 Electromagnetic Interference (EMI)

Based on a simple application running on the product (toggling 2 LEDs through the I/O ports), the product is monitored in terms of emission. This emission test is in line with the norm SAE J 1752/3 which specifies the board and the loading of each pin.

Symbol	Parameter	Conditions	Monitored Frequency Band	Max vs. [f_{OSC}/f_{CPU}]		Unit
				6/4MHz	12/8MHz	
S_{EMI}	Peak level	V_{DD}=5V, T_A=+25°C, PDIP42 package, conforming to SAE J 1752/3	0.1MHz to 30MHz	35	38	dBµV
			30MHz to 130MHz	42	45	
			130MHz to 1GHz	28	32	
			SAE EMI Level	4	4.5	-

Notes:

1. Data based on characterization results, not tested in production.

12.7.3 Absolute Maximum Ratings (Electrical Sensitivity)

Based on three different tests (ESD, LU and DLU) using specific measurement methods, the product is stressed in order to determine its performance in terms of electrical sensitivity. For more details, refer to the application note AN1181.

12.7.3.1 Electrostatic Discharge (ESD)

Electrostatic Discharges (a positive then a negative pulse separated by 1 second) are applied to the pins of each sample according to each pin combination. The sample size depends on the number of supply pins in the device (3 parts*(n+1) supply pin). This test conforms to the JESD22-A114A/A115A standard.

Absolute Maximum Ratings

Symbol	Ratings	Conditions	Maximum value [1]	Unit
$V_{ESD(HBM)}$	Electrostatic discharge voltage (Human Body Model)	T_A=+25°C	2000	V

Notes:

1. Data based on characterization results, not tested in production.

12.7.3.2 Static latchup

3 complementary static tests are required on 10 parts to assess the latchup performance. A supply overvoltage (applied to each power supply pin) and a current injection (applied to each input, output and configurable I/O pin) are performed on each sample. This test conforms to the EIA/JESD 78 IC latchup standard. For more details, refer to the application note AN1181.

Electrical Sensitivities

Symbol	Parameter	Conditions	Class [1]
LU	Static latchup class	T_A=+25°C	A

Notes:

1. Class description: A Class is an STMicroelectronics internal specification. All its limits are higher than the JEDEC specifications, that means when a device belongs to Class A it exceeds the JEDEC standard. B Class strictly covers all the JEDEC criteria (international standard).

12.8 I/O PORT PIN CHARACTERISTICS

12.8.1 General Characteristics

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage					$0.3xV_{DD}$	V
V_{IH}	Input high level voltage			$0.7xV_{DD}$			
V_{IN}	Input voltage	True Open Drain I/O pins		V_{SS}		6.0	V
		Other I/O pins				V_{DD}	
V_{hys}	Schmitt trigger voltage hysteresis				400		mV
I_L	Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$				±1	µA
I_S	Static current consumption [1]	Floating input mode			400		
R_{PU}	Weak pull-up equivalent resistor [2]	$V_{IN}=V_{SS}$	V_{DD}=5V	50	80	150	kΩ
C_{IO}	I/O pin capacitance				5		pF
$t_{f(IO)out}$	Output high to low level fall time	C_L=50pF Between 10% and 90%			25		ns
$t_{r(IO)out}$	Output low to high level rise time				25		
$t_{w(IT)in}$	External interrupt pulse time [3]			1			t_{CPU}

Figure 61. Two typical Applications with unused I/O Pin

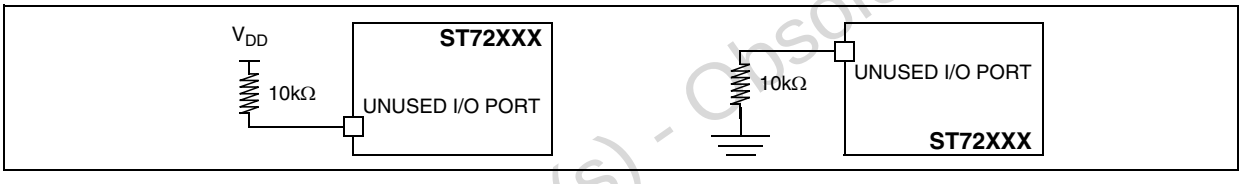


Figure 62. Typical I_{PU} vs. V_{DD} with $V_{IN}=V_{SS}$

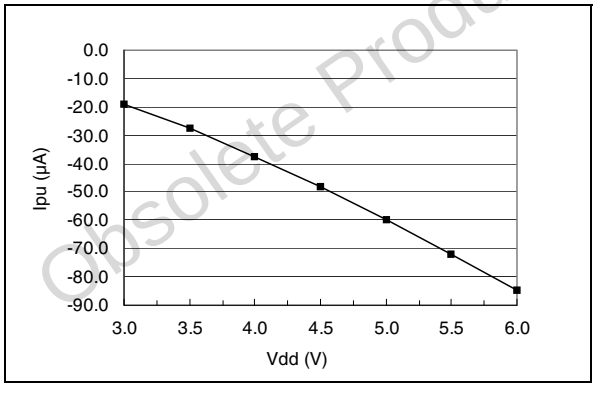
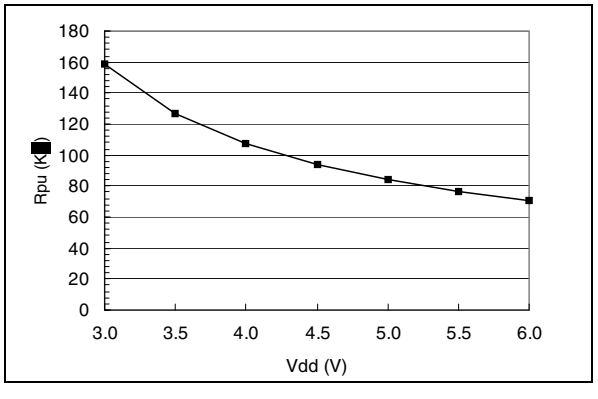


Figure 63. Typical R_{PU} vs. V_{DD} with $V_{IN}=V_{SS}$



Notes:

1. Configuration not recommended, all unused pins must be kept at a fixed voltage: using the output mode of the I/O for example or an external pull-up or pull-down resistor (see Figure 61). Static peak current value taken at a fixed V_{IN} value, based on design simulation and technology characteristics, not tested in production. This value depends on V_{DD} and temperature values.

2. The R_{PU} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{PU} current characteristics described in Figure 62). This data is based on characterization results.

3. To generate an external interrupt, a minimum pulse width has to be applied on an I/O port pin configured as an external interrupt source.

12.8.2 Output Driving Current

Subject to general operating condition for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions		Min	Max	Unit
V_{OL} [1]	Output low level voltage for a standard I/O pin when up to 8 pins are sunk at same time (see Figure 64)	V_{DD}=5V	I_{IO}=+5mA		1.3	V
			I_{IO}=+2mA		0.4	
	Output low level voltage for a high sink I/O pin when up to4 pins are sunk at same time (see Figure 65)		I_{IO}=+20mA		1.3	
			I_{IO}=+8mA		0.4	
V_{OH} [2]	Output high level voltage for an I/O pin when up to 8 pins are sourced at same time (see Figure 66)		I_{IO}=-5mA	V_{DD}-2.0		
			I_{IO}=-2mA	V_{DD}-0.8		

Figure 64. Typ. V_{OL} at V_{DD}=5V (std. port)

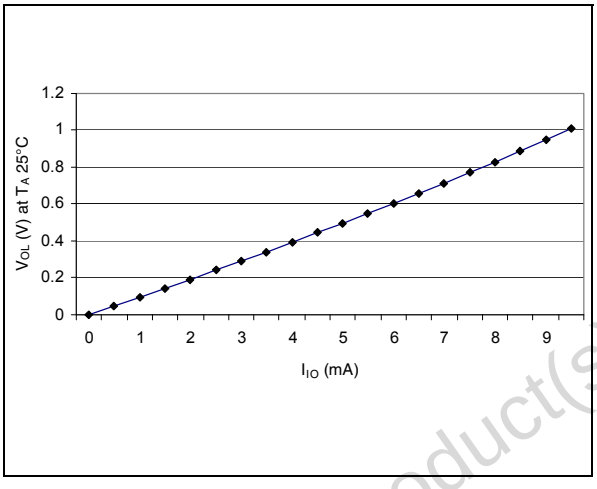


Figure 65. Typ. V_{OL} at V_{DD}=5V (high-sink)

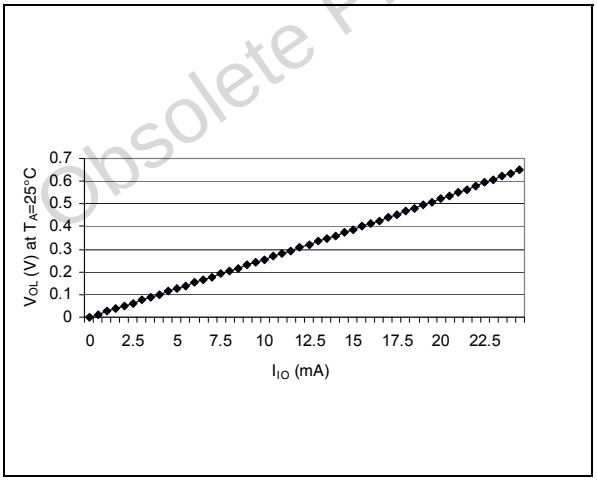


Figure 66. Typ. V_{DD}-V_{OH} at V_{DD}=5V (std. port)

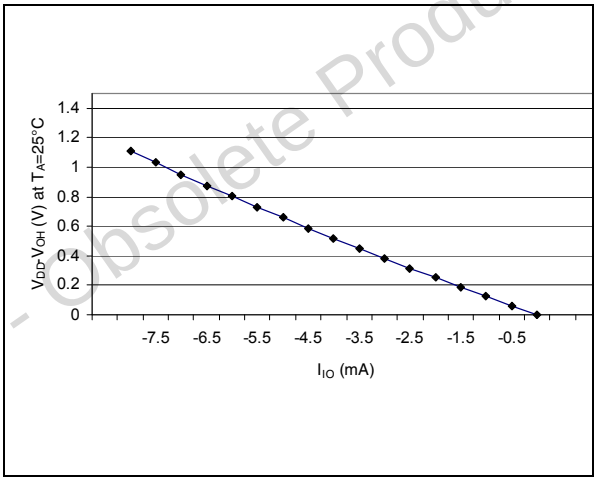
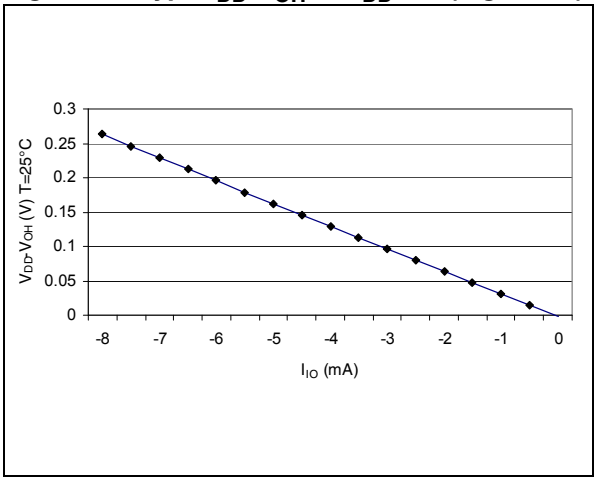


Figure 67. Typ. V_{DD}-V_{OH} at V_{DD}=5V (high-sink)



Notes:

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 12.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.

2. The I_{IO} current sourced must always respect the absolute maximum rating specified in Section 12.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}. True open drain I/O pins does not have V_{OH}.

Figure 68. Typical V_{OL} vs. V_{DD} (standard port)

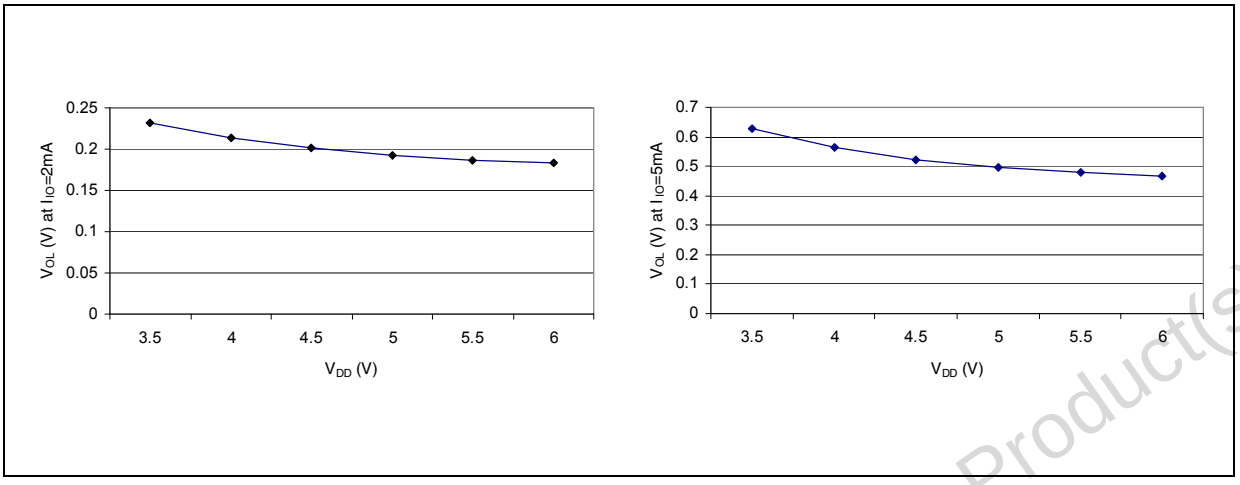


Figure 69. Typical V_{OL} vs. V_{DD} (high-sink port)

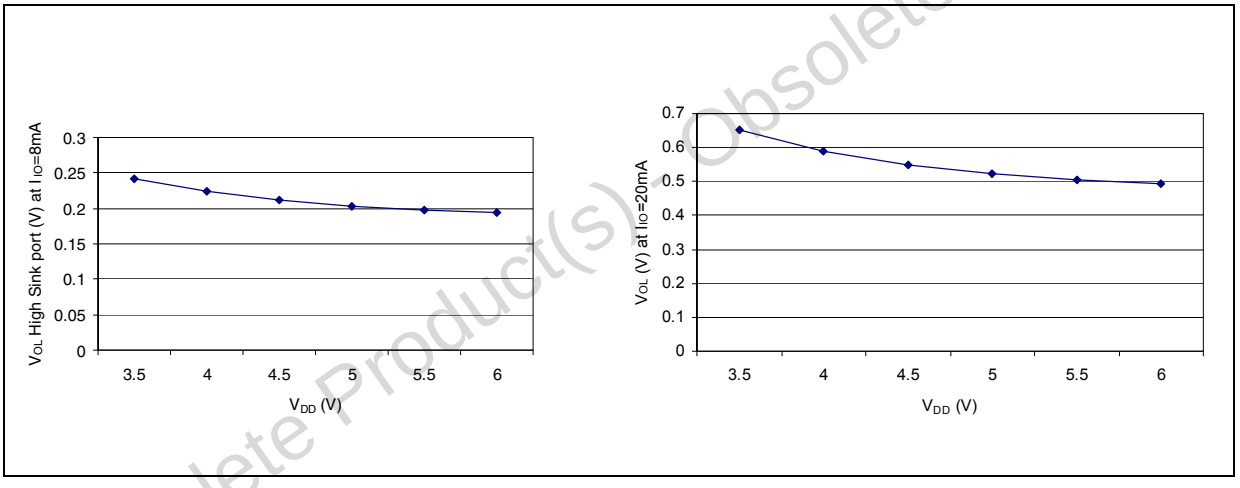


Figure 70. Typical V_{DD}-V_{OH} vs. V_{DD} (standard port)

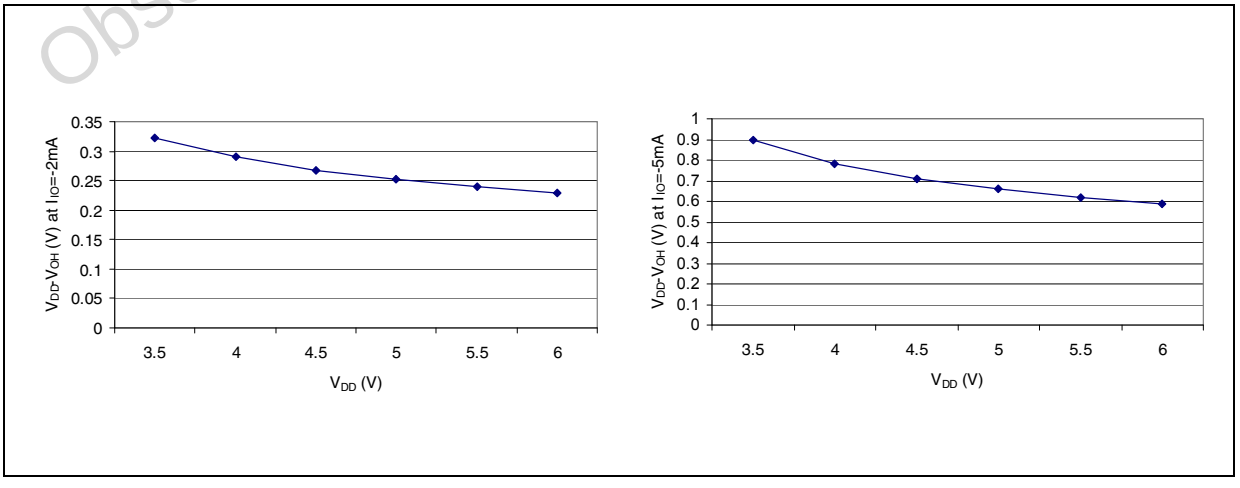
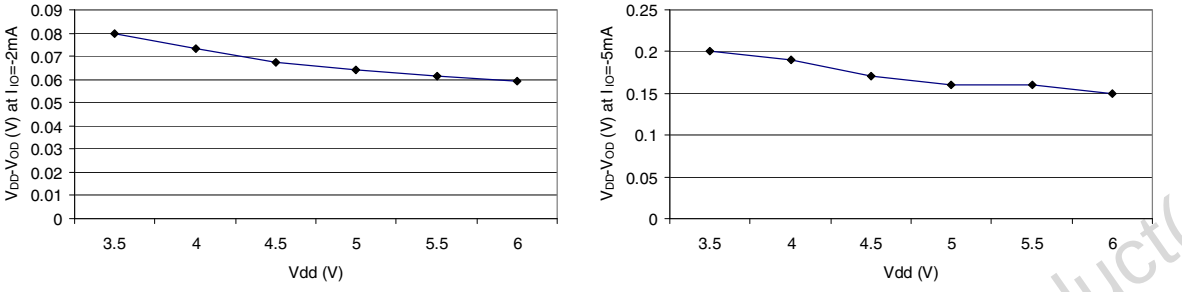


Figure 71. Typical V_{DD}-V_{OH} vs. V_{DD} (high sink port)



12.9 CONTROL PIN CHARACTERISTICS

12.9.1 Asynchronous $\overline{\text{RESET}}$ Pin

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions		Min	Typ [1]	Max	Unit
V_{IH}	Input High Level Voltage			0.7xV_{DD}		V_{DD}	V
V_{IL}	Input Low Voltage			V_{SS}		0.3xV_{DD}	V
V_{hys}	Schmitt trigger voltage hysteresis [3]				400		mV
V_{OL}	Output low level voltage [4] (see Figure 73, Figure 74)	V_{DD}=5V	I_{IO}=5mA			1 [2]	V
			I_{IO}=2mA			0.4 [2]	
R_{ON}	Weak pull-up equivalent resistor [5]	V_{IN}=V_{SS}			60		kΩ
$t_{w(RSTL)out}$	Generated reset pulse duration	External pin or internal reset sources			6 30		$1/f_{SFOSC}$ µs
$t_{h(RSTL)in}$	External reset pulse hold time [6]			10			µs

Notes:

1. Unless otherwise specified, typical data are based on T_A=25°C and V_{DD}=5V, not tested in production.
2. Data guaranteed by design.
3. Hysteresis voltage between Schmitt trigger switching levels. Based on characterization results, not tested.
4. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 12.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.
5. The R_{ON} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{ON} current characteristics described in Figure 72). This data is based on characterization results, not tested in production.
6. To guarantee the reset of the device, a minimum pulse has to be applied to $\overline{\text{RESET}}$ pin. All short pulses applied on $\overline{\text{RESET}}$ pin with a duration below $t_{h(RSTL)in}$ can be ignored.

Figure 72. Typical I_{ON} vs. V_{DD} with $V_{IN}=V_{SS}$

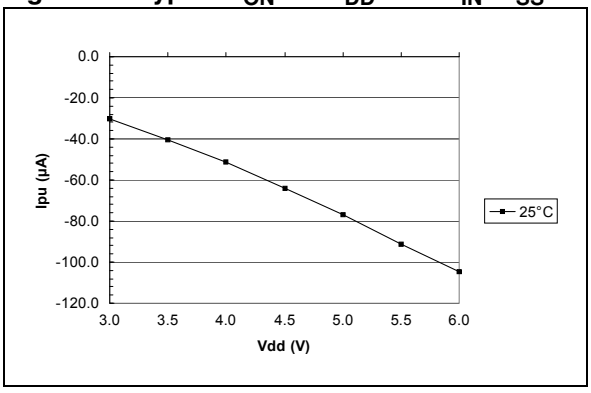


Figure 73. Typical V_{OL} at V_{DD}=5V ($\overline{RESET}$)

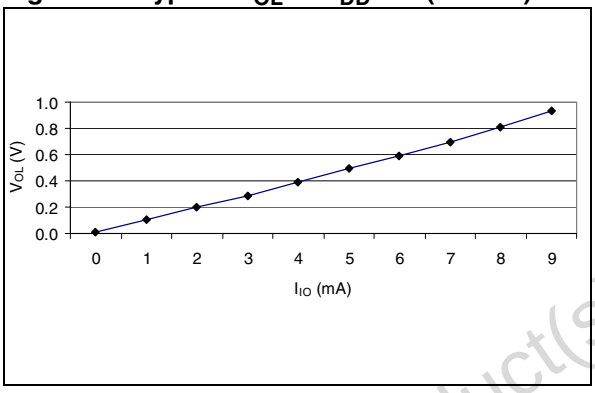


Figure 74. Typical V_{OL} vs. V_{DD} ($\overline{RESET}$)

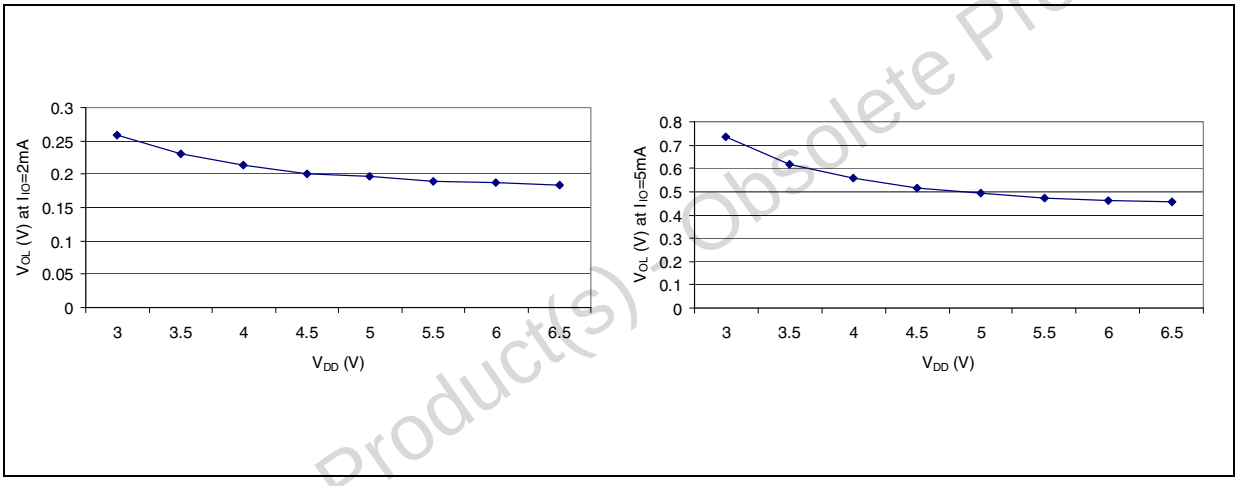


Figure 75. $\overline{\text{RESET}}$ pin protection when LVD is enabled.[1)2)3)4)]

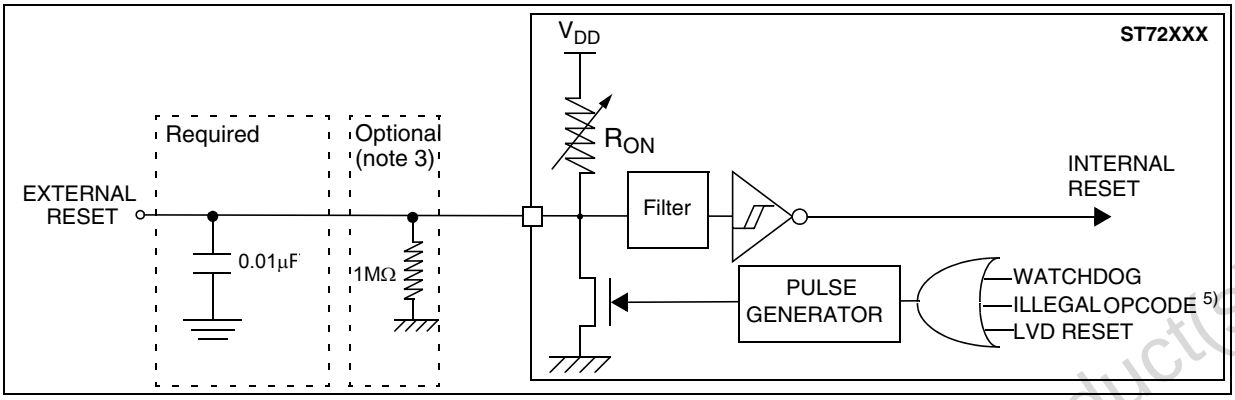
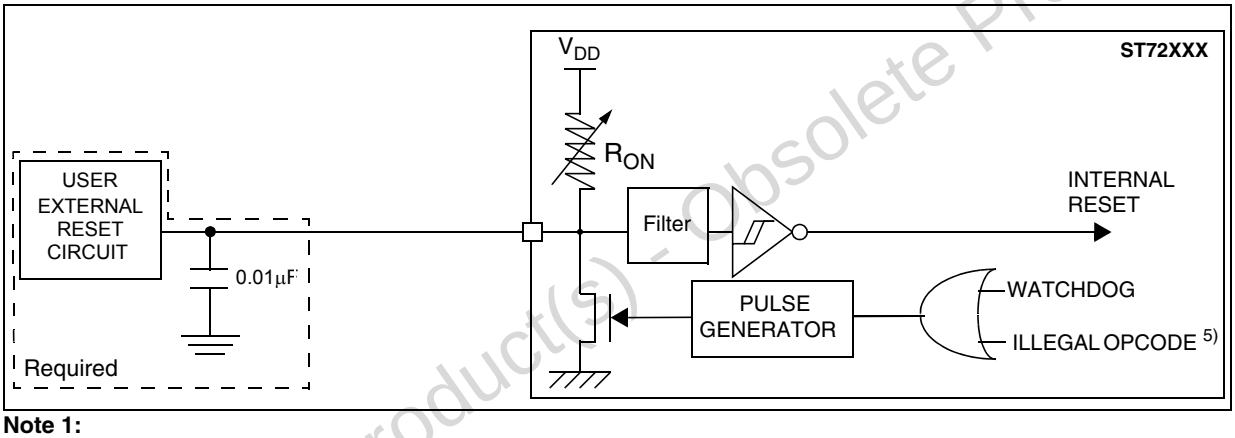


Figure 76. $\overline{\text{RESET}}$ pin protection when LVD is disabled.[1)]



Note 1:

- The reset network protects the device against parasitic resets.
- The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).
- Whatever the reset source is (internal or external), the user must ensure that the level on the $\overline{\text{RESET}}$ pin can go below the V_{IL} max. level specified in section 12.9.1 on page 113. Otherwise the reset will not be taken into account internally.
- Because the reset circuit is designed to allow the internal RESET to be output in the $\overline{\text{RESET}}$ pin, the user must ensure that the current sunk on the $\overline{\text{RESET}}$ pin is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in section 12.2.2 on page 102.

Note 2: When the LVD is enabled, it is recommended not to connect a pull-up resistor or capacitor. A 10nF pull-down capacitor is required to filter noise on the reset line.

Note 3: In case a capacitive power supply is used, it is recommended to connect a 1MΩ pull-down resistor to the $\overline{\text{RESET}}$ pin to discharge any residual voltage induced by the capacitive effect of the power supply (this will add 5µA to the power consumption of the MCU).

Note 4: Tips when using the LVD:

- 1. Check that all recommendations related to the reset circuit have been applied (see notes above).
- 2. Check that the power supply is properly decoupled (100nF + 10µF close to the MCU). Refer to AN1709 and AN2017. If this cannot be done, it is recommended to put a 100nF + 1MΩ pull-down on the $\overline{\text{RESET}}$ pin.
- 3. The capacitors connected on the RESET pin and also the power supply are key to avoid any start-up marginality. In most cases, steps 1 and 2 above are sufficient for a robust solution. Otherwise: replace 10nF pull-down on the $\overline{\text{RESET}}$ pin with a 5µF to 20µF capacitor."

12.10 TIMER PERIPHERAL CHARACTERISTICS

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (output compare, input capture, external clock, PWM output...).

12.10.1 8-Bit PWM-ART Auto-Reload Timer

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{res(PWM)}$	PWM resolution time		1			t_{CPU}
		f_{CPU}=8MHz	125			ns
f_{EXT}	ART external clock frequency		0		$f_{CPU}/2$	MHz
f_{PWM}	PWM repetition rate		0		$f_{CPU}/2$	
Res_{PWM}	PWM resolution				8	bit
V_{OS}	PWM/DAC output step voltage	V_{DD}=5V, Res=8 bits		20		mV

12.11 COMMUNICATION INTERFACE CHARACTERISTICS

12.11.1 USB - Universal Bus Interface

(Operating conditions T_A = 0 to +70°C, V_{DD} = 4.0 to 5.25V unless otherwise specified)

USB DC Electrical Characteristics					
Parameter	**Symbol**	**Conditions** [2]	**Min.**	**Max.**	**Unit**
Differential Input Sensitivity	VDI	I(D+, D-)	0.2 [3]		V
Differential Common Mode Range	VCM	Includes VDI range	0.8 [3]	2.5 [3]	V
Single Ended Receiver Threshold	VSE		0.8 [3]	2.0 [3]	V
Static Output Low	VOL	RL of 1.5K ohms to 3.6V [1]		0.3	V
Static Output High	VOH	RL of 15K ohms to V_{SS} [1]	2.8	3.6	V
USBVCC: voltage level [4]	USBV	V_{DD}=5V	3.00	3.60	V

Notes

1. RL is the load connected on the USB drivers.
2. All the voltages are measured from the local ground potential.
3. Not tested in production, guaranteed by design.
4. To improve EMC performance (noise immunity), it is recommended to connect a 100nF capacitor to the USBVCC pin.

Figure 77. USB: Data Signal Rise and Fall Time

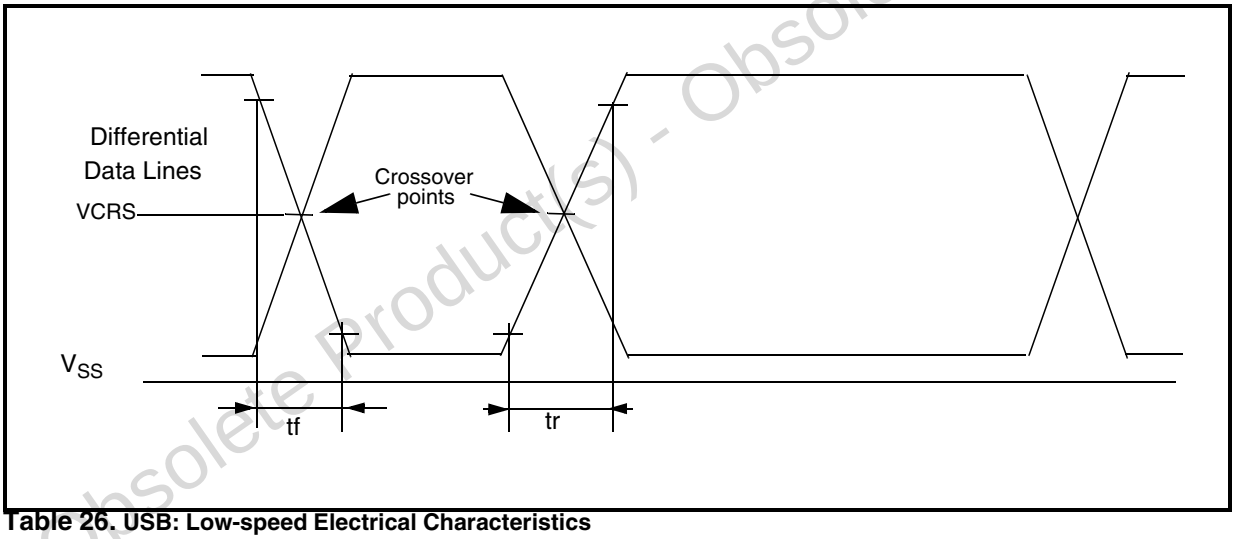


Table 26. USB: Low-speed Electrical Characteristics

Parameter	Symbol	Conditions	Min	Max	Unit
Driver characteristics:					
Rise time	tr	CL=50 pF [1]	75		ns
		CL=600 pF [1]		300	ns
Fall Time	tf	CL=50 pF [1]	75		ns
		CL=600 pF [1]		300	ns
Rise/ Fall Time matching	trfm	tr/tf	80	120	%
Output signal Crossover Voltage	VCRS		1.3	2.0	V

Note 1: Measured from 10% to 90% of the data signal. For more detailed informations, please refer to Chapter 7 (Electrical) of the USB specification (version 1.1).

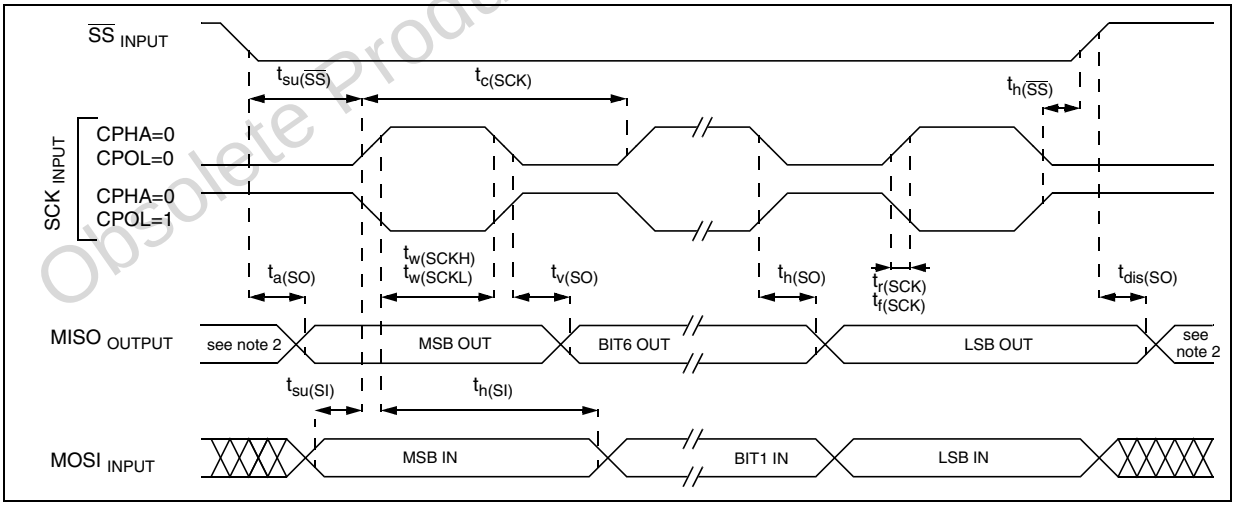
12.11.2 SPI - Serial Peripheral Interface

Subject to general operating condition for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Refer to I/O port characteristics for more details on the input/output alternate function characteristics ($\overline{SS}$, SCK, MOSI, MISO).

Symbol	Parameter	Conditions	Min [1]	Max [1]	Unit
f_{SCK} $1/t_{c(SCK)}$	SPI clock frequency	Master f_{CPU}=8MHz	$f_{CPU}/128$ 0.0625	$f_{CPU}/4$ 2	MHz
		Slave f_{CPU}=8MHz	0	$f_{CPU}/2$ 4	
$t_{r(SCK)}$ $t_{f(SCK)}$	SPI clock rise and fall time		see I/O port pin description		
$t_{su(\overline{SS})}$	$\overline{SS}$ setup time	Slave	120		ns
$t_{h(\overline{SS})}$	$\overline{SS}$ hold time	Slave	120		
$t_{w(SCKH)}$ $t_{w(SCKL)}$	SCK high and low time	Master Slave	100 90		
$t_{su(MI)}$ $t_{su(SI)}$	Data input setup time	Master Slave	100 100		
$t_{h(MI)}$ $t_{h(SI)}$	Data input hold time	Master Slave	100 100		
$t_{a(SO)}$	Data output access time	Slave	0	120	
$t_{dis(SO)}$	Data output disable time	Slave		240	
$t_{v(SO)}$	Data output valid time	Slave (after enable edge)		120	
$t_{h(SO)}$	Data output hold time		0		
$t_{v(MO)}$	Data output valid time	Master (after enable edge)		120	
$t_{h(MO)}$	Data output hold time		0		

Figure 78. SPI Slave Timing Diagram with CPHA=0 [3]



Notes:

1. Data based on design simulation and/or characterization results, not tested in production.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends on the I/O port configuration.
3. Measurement points are done at CMOS levels: $0.3xV_{DD}$ and $0.7xV_{DD}$.

Figure 79. SPI Slave Timing Diagram with CPHA=1[1)]

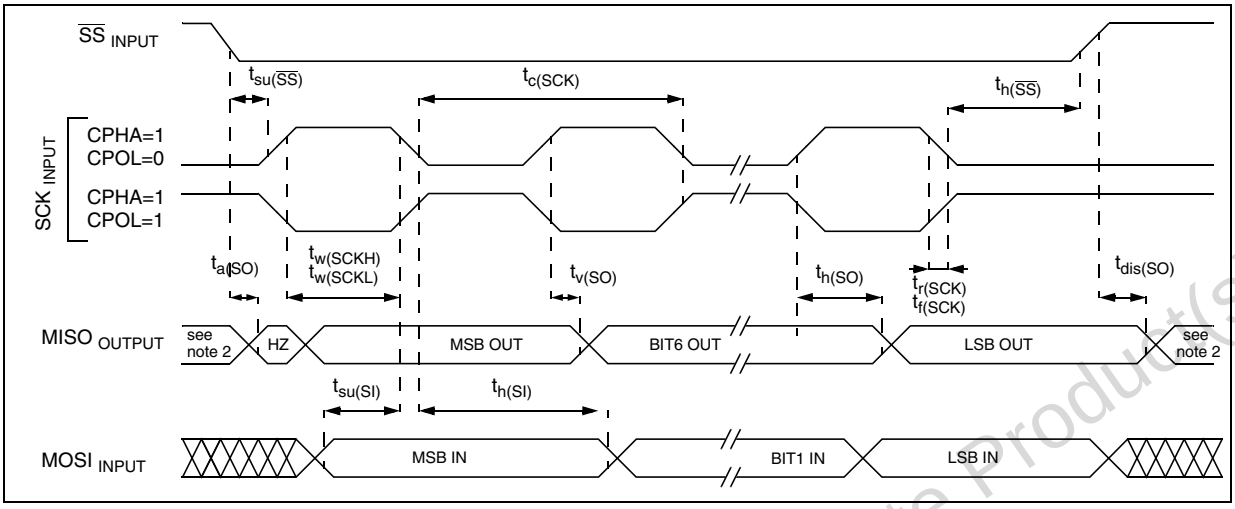
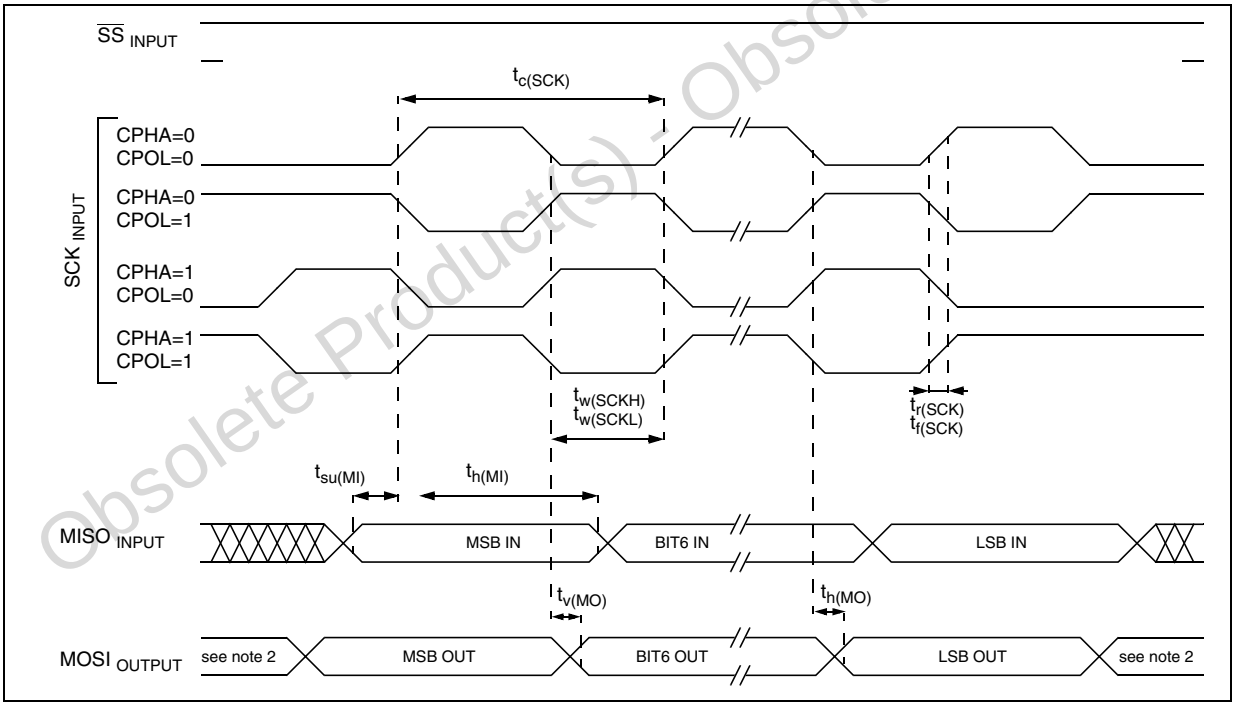


Figure 80. SPI Master Timing Diagram [1)]



Notes:

1. Measurement points are done at CMOS levels: $0.3xV_{DD}$ and $0.7xV_{DD}$.

2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends of the I/O port configuration.

12.12 10-BIT ADC CHARACTERISTICS

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
f_{ADC}	ADC clock frequency		0.4		4	MHz
V_{AIN}	Conversion voltage range [2]		V_{SSA}		V_{DDA}	V
R_{AIN}	External input impedance				see Figure 81 and Figure 82 [3][4][5]	kΩ
C_{AIN}	External capacitor on analog input					pF
f_{AIN}	Variation frequency of analog input signal					Hz
C_{ADC}	Internal sample and hold capacitor			6		pF
t_{CONV} [6]	Conversion time Flash silicon rev. G devices	f_{ADC}=4MHz		4		µs
				16		$1/f_{ADC}$
	Conversion time Flash silicon rev. X and ROM rev. Z devices			28		µs
				112		$1/f_{ADC}$

Figure 81. R_{AIN} max. vs f_{ADC} with C_{AIN}=0pF [4]

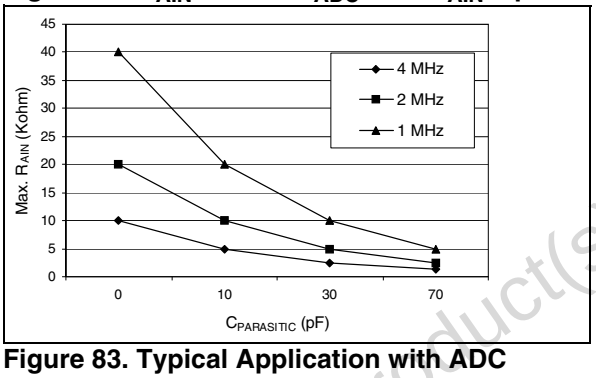


Figure 82. Recommended C_{AIN}/R_{AIN} values[5]

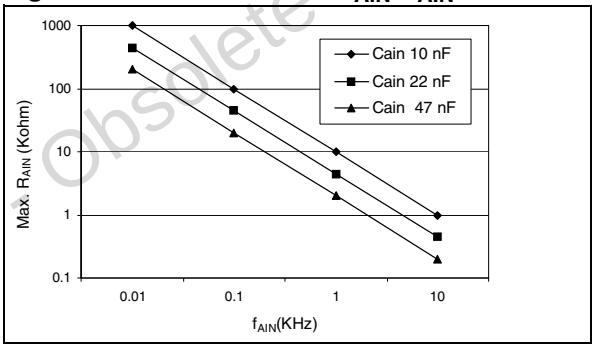
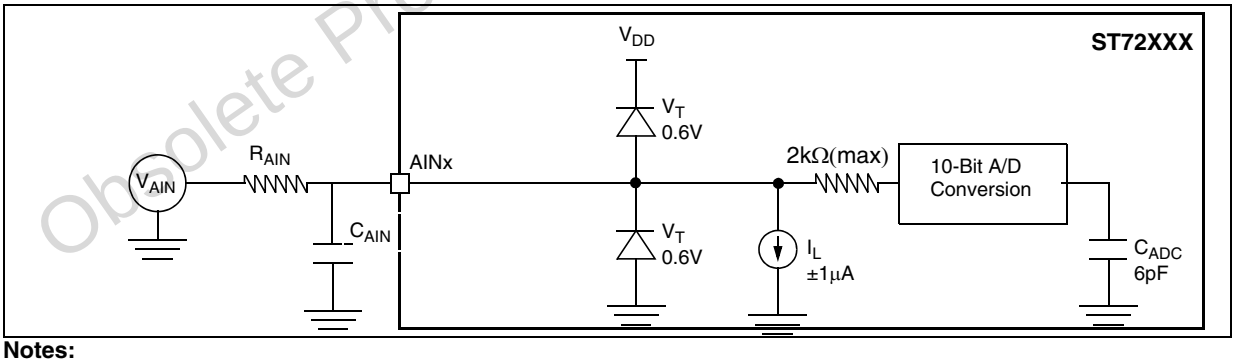


Figure 83. Typical Application with ADC



Notes:

1. Unless otherwise specified, typical data are based on T_A=25°C and V_{DD}-V_{SS}=5V. They are given only as design guidelines and are not tested.

2. When V_{DDA} and V_{SSA} pins are not available on the pinout, the ADC refers to V_{DD} and V_{SS}.

3. Any added external serial resistor will downgrade the ADC accuracy (especially for resistance greater than 10kΩ). Data based on characterization results, not tested in production.

4. $C_{PARASITIC}$ represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (3pF). A high $C_{PARASITIC}$ value will downgrade conversion accuracy. To remedy this, f_{ADC} should be reduced.

5. This graph shows that depending on the input signal variation (f_{AIN}), C_{AIN} can be increased for stabilization and to allow the use of a larger serial resistor (R_{AIN}). It is valid for all f_{ADC} frequencies ≤ 4MHz.

6. Please refer to Important Notes on conversion speed, Section 15.2 and also to Figure 92. on page 137 for details on silicon revision identification.

12.12.0.1 Analog Power Supply and Reference Pins

Depending on the MCU pin count, the package may feature separate V_{DDA} and V_{SSA} analog power supply pins. These pins supply power to the A/D converter cell and function as the high and low reference voltages for the conversion. In some packages V_{DDA} and V_{SSA} pins are not available (refer to Table 1, "Device Pin Description," on page 8). In this case the analog supply and reference pads are internally bonded to the V_{DD} and V_{SS} pins.

Separation of the digital and analog power pins allow board designers to improve A/D performance. Conversion accuracy can be impacted by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines (see Section 10.7.3.2 PCB Design Guidelines).

12.12.0.2 General PCB Design Guidelines

To obtain best results, some general design and layout rules should be followed when designing the application PCB to shield the noise-sensitive, analog physical interface from noise-generating CMOS logic signals.

– Use separate digital and analog planes. The analog ground plane should be connected to the digital ground plane via a single point on the PCB.

– Filter power to the analog power planes. The best solution is to connect capacitors, with good high frequency characteristics, between the power and ground lines, placing 0.1µF and 10pF capacitors as close as possible to the ST7 power supply pins and a 10µF capacitor close to the power source (see Figure 84).

– The analog and digital power supplies should be connected in a star network. Do not use a resistor, as V_{DDA} is used as a reference voltage by the A/D converter and any resistance would cause a voltage drop and a loss of accuracy.

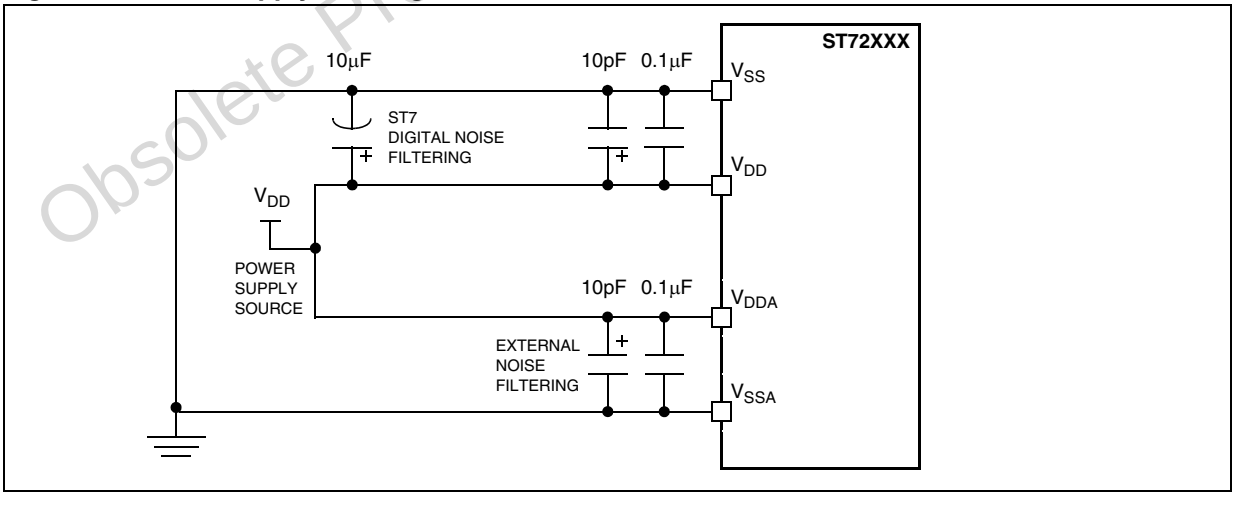
– Properly place components and route the signal traces on the PCB to shield the analog inputs. Analog signals paths should run over the analog ground plane and be as short as possible. Isolate analog signals from digital signals that may switch while the analog inputs are being sampled by the A/D converter. Do not toggle digital outputs on the same I/O port as the A/D input being converted.

12.12.0.3 Specific Application Design Guidelines

– When a USB transmission is taking place during A/D conversion, the noise caused on the analog power supply by the USB transmission may result in a loss of ADC accuracy.

– If the USB is used to supply power to the application, this causes noise which may result in a loss of ADC accuracy.

Figure 84. Power Supply Filtering



12.12.1 ADC Accuracy

Table 27. f_{CPU}=8 MHz, f_{ADC}=4 MHz R_{AIN}< 10kΩ 2)

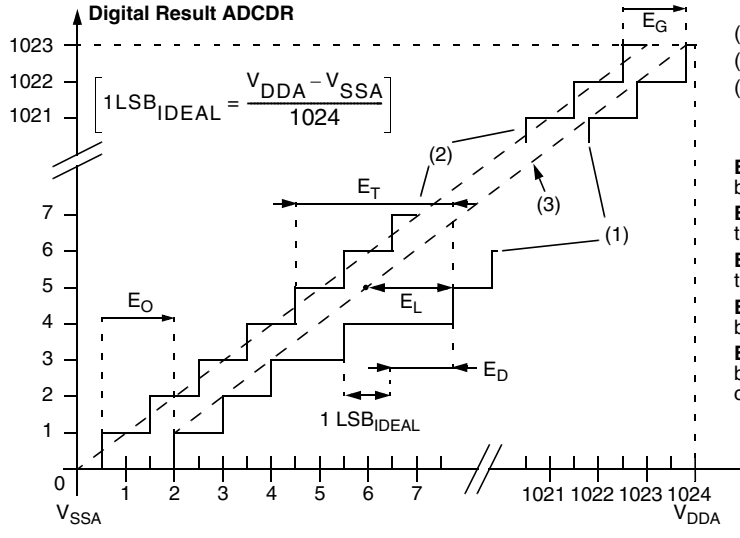
Symbol	Parameter	Conditions	Typ	Max 1)	Unit
$\|E_T\|$	Total unadjusted error	V_{DD}= 4V-5.5V	3		LSB
$\|E_O\|$	Offset error		1	2	
$\|E_G\|$	Gain Error		0.7	2	
$\|E_D\|$	Differential linearity error		1.3	2	
$\|E_L\|$	Integral linearity error		2.9	5	

Notes:

1. Not tested in production, guaranteed by characterization. All accuracy measurements are taken with the MCU in WAIT mode (no I/O switching) and when adequate low-pass filtering is present (0.1 µF capacitor between V_{DD}/V_{DDA} and V_{SS}/V_{SSA}). Outside these conditions, a degree of microcontroller noise may result, causing accuracy errors which will vary based on board layout and the type of CPU activity.

2. ADC Accuracy vs. Negative Injection Current:
Injecting negative current on any of the analog input pins significantly reduces the accuracy of the conversion being performed on another analog input.
For I_{INJ-}=0.8mA, the typical leakage induced inside the die is 1.6µA and the effect on the ADC accuracy is a loss of 4 LSB for each 10KΩ increase of the external analog source impedance. It is recommended to add a Schottky diode (pin to ground) to analog pins which may potentially inject negative current. Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in Section 12.8 does not affect the ADC accuracy.

Figure 85. ADC Accuracy Characteristics



(1) Example of an actual transfer curve
(2) The ideal transfer curve
(3) End point correlation line

E_T=Total Unadjusted Error: maximum deviation between the actual and the ideal transfer curves.

E_O=Offset Error: deviation between the first actual transition and the first ideal one.

E_G=Gain Error: deviation between the last ideal transition and the last actual one.

E_D=Differential Linearity Error: maximum deviation between actual steps and the ideal one.

E_L=Integral Linearity Error: maximum deviation between any actual transition and the end point correlation line.

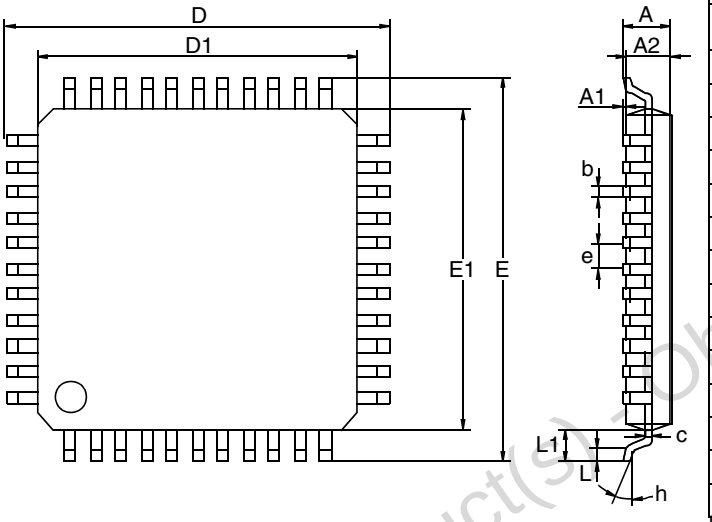
13 PACKAGE CHARACTERISTICS

In order to meet environmental requirements, ST offers this device in different grades of ECOPACK® packages, depending on their level of environmental compliance. ECOPACK® specifications, grade definitions and product status are available at: www.st.com.

ECOPACK® is an ST trademark.

13.1 PACKAGE MECHANICAL DATA

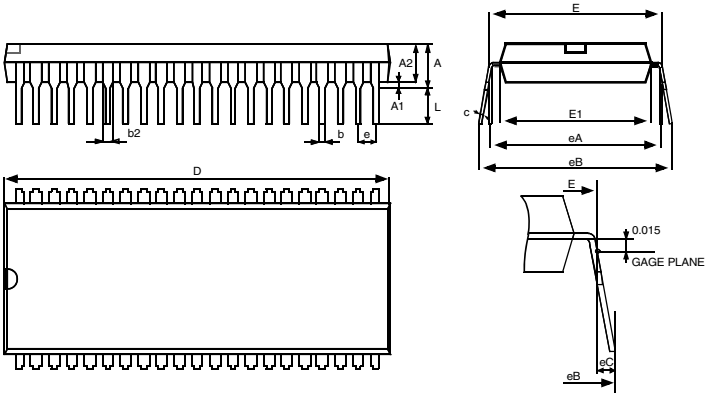
Figure 86. 44-Pin Low Profile Quad Flat Package (10x10)



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.0630
A1	0.05		0.15	0.0020		0.0059
A2	1.35	1.40	1.45	0.0531	0.0551	0.0571
b	0.30	0.37	0.45	0.0118	0.0146	0.0177
C	0.09		0.20	0.0035		0.0079
D		12.00			0.4724	
D1		10.00			0.3937	
E		12.00			0.4724	
E1		10.00			0.3937	
e		0.80			0.0315	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.0177	0.0236	0.0295
L1		1.00			0.0394	
	Number of Pins					
N	44					

Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.

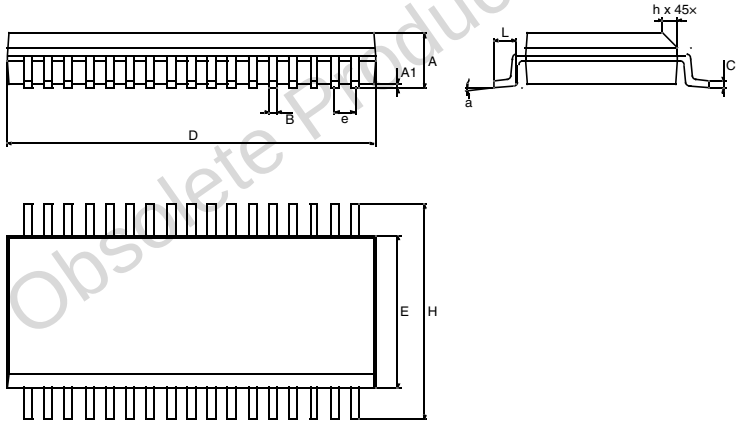
Figure 87. 42-Pin Plastic Dual In-Line Package, Shrink 600-mil Width



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			5.08			0.2000
A1	0.51			0.0201		
A2	3.05	3.81	4.57	0.1201	0.1500	0.1799
b	0.38	0.46	0.56	0.0150	0.0181	0.0220
b2	0.89	1.02	1.14	0.0350	0.0402	0.0449
c	0.23	0.25	0.38	0.0091	0.0098	0.0150
D	36.58	36.83	37.08	1.4402	1.4500	1.4598
E	15.24		16.00	0.6000		0.6299
E1	12.70	13.72	14.48	0.5000	0.5402	0.5701
e		1.78			0.0701	
eA		15.24			0.6000	
eB			18.54			0.7299
eC			1.52			0.0598
L	2.54	3.30	3.56	0.1000	0.1299	0.1402
	Number of Pins					
N	42					

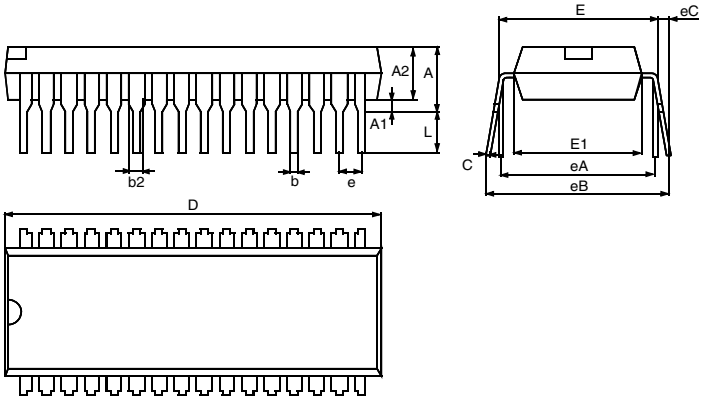
Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.

Figure 88. 34-Pin Plastic Small Outline Package, Shrink 300-mil Width



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A	2.464		2.642	0.097		0.104
A1	0.127		0.292	0.005		0.012
B	0.356		0.483	0.014		0.019
C	0.231		0.318	0.009		0.013
D	17.729		18.059	0.698		0.711
E	7.417		7.595	0.292		0.299
e		1.016			0.040	
H	10.160		10.414	0.400		0.410
h	0.635		0.737	0.025		0.029
α	0°		8°	0°		8°
L	0.610		1.016	0.024		0.040
	Number of Pins					
N	34					

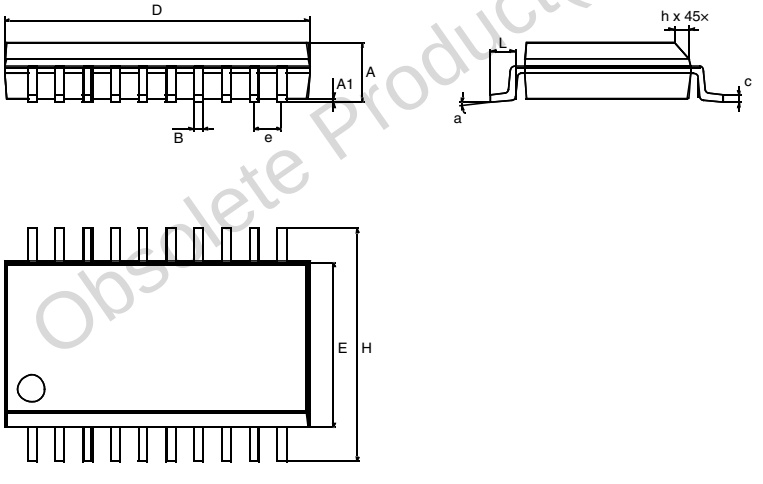
Figure 89. 32-Pin Plastic Dual In-Line Package, Shrink 400-mil Width



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A	3.56	3.76	5.08	0.1402	0.1480	0.2000
A1	0.51			0.0201		
A2	3.05	3.56	4.57	0.1201	0.1402	0.1799
b	0.36	0.46	0.58	0.0142	0.0181	0.0228
b1	0.76	1.02	1.40	0.0299	0.0402	0.0551
C	0.20	0.25	0.36	0.0079	0.0098	0.0142
D	27.43		28.45	1.0799		1.1201
E	9.91	10.41	11.05	0.3902	0.4098	0.4350
E1	7.62	8.89	9.40	0.3000	0.3500	0.3701
e		1.78			0.0701	
eA		10.16			0.4000	
eB			12.70			0.5000
eC			1.40			0.0551
L	2.54	3.05	3.81	0.1000	0.1201	0.1500
	Number of Pins					
N	32					

Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.

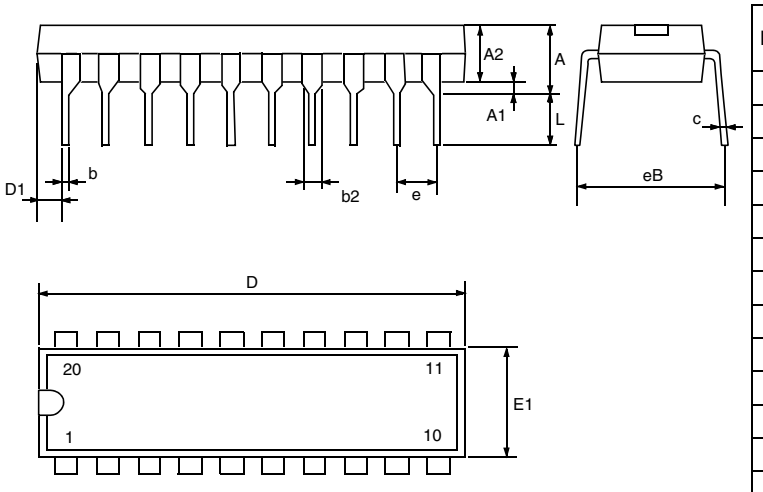
Figure 90. 20-Pin Plastic Small Outline Package, 300-mil Width



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A	2.35		2.65	0.0925		0.1043
A1	0.10		0.30	0.0039		0.0118
B	0.33		0.51	0.0130		0.0201
C	0.23		0.32	0.0091		0.0126
D	12.60		13.00	0.4961		0.5118
E	7.40		7.60	0.2913		0.2992
e		1.27			0.0500	
H	10.00		10.65	0.3937		0.4193
h	0.25		0.75	0.0098		0.0295
α	0°		8°	0°		8°
L	0.40		1.27	0.0157		0.0500
	Number of Pins					
N	20					

Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.

Figure 91. 20-Pin Plastic Dual In-Line Package, 300-mil Width



Dim.	mm			inches[1)]		
	Min	Typ	Max	Min	Typ	Max
A			5.33			0.2098
A1	0.38			0.0150		
A2	2.92	3.30	4.95	0.1150	0.1299	0.1949
b	0.36	0.46	0.56	0.0142	0.0181	0.0220
b2	1.14	1.52	1.78	0.0449	0.0598	0.0701
c	0.20	0.25	0.36	0.0079	0.0098	0.0142
D	24.89	26.16	26.92	0.9799	1.0299	1.0598
D1	0.13			0.0051		
e		2.54			0.1000	
eB			10.92			0.4299
E1	6.10	6.35	7.11	0.2402	0.2500	0.2799
L	2.92	3.30	3.81	0.1150	0.1299	0.1500
	Number of Pins					
N	20					

Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.

14 DEVICE CONFIGURATION AND ORDERING INFORMATION

Each device is available for production in user programmable versions (FLASH) as well as in factory coded versions (ROM).

ST7262 devices are ROM versions.

ST72F62 FLASH devices are shipped to customers with a default content (FFh). This implies that FLASH devices have to be configured by the customer using the Option Byte while the ROM devices are factory-configured.

14.1 OPTION BYTE

The Option Byte allows the hardware configuration of the microcontroller to be selected.
The Option Byte has no address in the memory map and can be accessed only in programming mode using a standard ST7 programming tool. The default content of the FLASH is fixed to FFh. This means that all the options have “1” as their default value.

7							0
-	-	WDG SW	NEST	LVD	-	OSC 12/6	FMP_ R

Bits 7:6 = Reserved.

Bit 5 = **WDGSW** *Hardware or software watchdog*
This option bit selects the watchdog type.
0: Hardware enabled
1: Software enabled

Bit 4 = **NEST**
This option bit selects the nested interrupts feature.
0: Nested interrupt feature disabled
1: Nested interrupt feature enabled

Bit 3 = **LVD** *Low Voltage Detector selection*
This option bit selects the LVD.
0: LVD enabled
1: LVD disabled

Bit 2= Reserved.

Bit 1 = **OSC12/6** *Oscillator selection*
This option bit selects the clock divider used to drive the USB interface at 6MHz.
0: 6 MHz oscillator (no divider for USB)
1: 12 Mhz oscillator (2 divider for USB)

Bit 0 = **FMP_R** *Memory Readout Protection*
Readout protection, when selected provides a protection against program memory content extraction and against write access to Flash memory. Erasing the option bytes when the FMP_R option is selected will cause the whole memory to be erased first and the device can be reprogrammed. Refer to the ST7 Flash Programming Reference Manual and section 4.3.1 on page 14 for more details.
0: Read-out protection enabled
1: Read-out protection disabled

14.2 DEVICE ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE

Customer code is made up of the ROM contents and the list of the selected options (if any). The ROM contents are to be sent on diskette, or by electronic means, with the hexadecimal file in .S19 format generated by the development tool. All unused bytes must be set to FFh.

The selected options are communicated to STMicroelectronics using the correctly completed OPTION LIST appended.
Refer to application note AN1635 for information on the counter listing returned by ST after code has been transferred.
The STMicroelectronics Sales Organization will be pleased to provide detailed information on contractual points.

Table 28. Supported part numbers

Part Number	Program Memory (Bytes)	RAM (Bytes)	Package
ST72F623F2B1	8K FLASH	384	PDIP20
ST72F623F2M1			SO20
ST72F622L2M1			SO34
ST72F621K4B1	16K FLASH	768	PDIP32
ST72F621L4M1			SO34
ST72F621J4B1	16K FLASH	768	PDIP42
ST72F621J4T1			LQFP44
ST72623F2B1	8K ROM	384	PDIP20
ST72623F2M1			SO20
ST72622L2M1			SO34
ST72621K4B1	16K ROM	768	PDIP32
ST72621L4M1			SO34
ST72621J4B1	16K ROM	768	PDIP42
ST72621J4T1			LQFP44

Contact ST sales office for product availability

14.3 DEVELOPMENT TOOLS

STMicroelectronics offers a range of hardware and software development tools for the ST7 microcontroller family. Full details of tools available for the ST7 from third party manufacturers can be obtain from the STMicroelectronics Internet site:
➟ http//mcu.st.com.

Tools from these manufacturers include C compliers, emulators and gang programmers.

STMicroelectronics Tools

Three types of development tool are offered by ST see Table 29 and Table 30 for more details.

Table 29. STMicroelectronics Tools Features

	In-Circuit Emulation	Programming Capability[1)]	Software Included
ST7 Emulator	Yes, powerful emulation features including trace/ logic analyzer	No	ST7 CD ROM with: – ST7 Assembly toolchain – STVD7 powerful Source Level Debugger for Win 3.1, Win 9x and NT – C compiler demo versions – Windows Programming Tools for Win 3.1, Win 9x and NT
ST7 Programming Board	No	Yes (All packages)	

Note:
1. In-Circuit Programming (ICP) interface for FLASH devices.

Table 30. Dedicated STMicroelectronics Development Tools

Supported Products	Evaluation Board	ST7 Emulator	ST7 Programming Board	Active Probe & Target Emulation Board
ST7262	ST7MDTULS-EVAL	ST7MDTU2-EMU2B	ST7MDTU2-EPB [1)]	ST7MDTU2-DBE2B

Note:
1. Add Suffix /EU or /US for the power supply for your region.

ST7262 MICROCONTROLLER OPTION LIST
(Last update: March 2006)

Customer: . . . . . .
Address: . . . . . .
. . . . . .
Contact: . . . . . .
Phone No: . . . . . .
Reference/ROM Code* : . . . . . .
*The ROM code name is assigned by STMicroelectronics.
ROM code must be sent in .S19 format. .Hex extension cannot be processed.

Device Type/Memory Size/Package (check only one option):

ROM DEVICE:	8K	16K
SDIP20:	[ ] ST72623F2B1	
SO20:	[ ] ST72623F2M1	
SDIP32:		[ ] ST72621K4B1
SO34:	[ ] ST72622L2M1	[ ] ST72621L4M1
SDIP42:		[ ] ST72621J4B1
LQFP44:		[ ] ST72621J4T1

DIE FORM:	8K	16K
20-pin:	[ ]	
32-pin:	[ ]	[ ]
34-pin:	[ ]	[ ]
42-pin:	[ ]	[ ]
44-pin:	[ ]	[ ]

Conditioning (check only one option):

Packaged Product (do not specify for DIP package)		Die Product (dice tested at 25°C only)
[ ] Tape & Reel	[ ] Tray (LQFP package only)	[ ] Tape & Reel
	[ ] Tube (SO package only)	[ ] Inked wafer
		[ ] Sawn wafer on sticky foil

Special Marking: [ ] No [ ] Yes "_ _ _ _ _ _ _ _ _ _ "
Authorized characters are letters, digits, '.', '-', '/' and spaces only.
Max character count: S020 (8 char. max) : _ _ _ _ _ _ _ _ S034 (13 char. max) : _ _ _ _ _ _ _ _ _ _ _ _ _
DIP20/DIP32/LQFP44 (10 char. max) : _ _ _ _ _ _ _ _ _ _ DIP42 (16 char. max) : _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Watchdog Selection:	[ ] Software activation	[ ] Hardware activation
Nested Interrupt:	[ ] Enabled	[ ] Disabled
LVD Reset :	[ ] Disabled	[ ] Enabled
Oscillator Selection :	[ ] 6 MHz.	[ ] 12 MHz.
Readout protection:	[ ] Enabled	[ ] Disabled

Date . . . . . Signature . . . . . .

Please download the latest version of this option list from:
http://www.st.com/mcu > downloads > ST7 microcontrollers > Option list

14.4 ST7 APPLICATION NOTES

Table 31. ST7 Application Notes

IDENTIFICATION	DESCRIPTION
APPLICATION EXAMPLES	
AN1658	SERIAL NUMBERING IMPLEMENTATION
AN1720	MANAGING THE READ-OUT PROTECTION IN FLASH MICROCONTROLLERS
AN1755	A HIGH RESOLUTION/PRECISION THERMOMETER USING ST7 AND NE555
AN1756	CHOOSING A DALI IMPLEMENTATION STRATEGY WITH ST7DALI
AN1812	A HIGH PRECISION, LOW COST, SINGLE SUPPLY ADC FOR POSITIVE AND NEGATIVE INPUT VOLTAGES
EXAMPLE DRIVERS	
AN 969	SCI COMMUNICATION BETWEEN ST7 AND PC
AN 970	SPI COMMUNICATION BETWEEN ST7 AND EEPROM
AN 971	I²C COMMUNICATION BETWEEN ST7 AND M24CXX EEPROM
AN 972	ST7 SOFTWARE SPI MASTER COMMUNICATION
AN 973	SCI SOFTWARE COMMUNICATION WITH A PC USING ST72251 16-BIT TIMER
AN 974	REAL TIME CLOCK WITH ST7 TIMER OUTPUT COMPARE
AN 976	DRIVING A BUZZER THROUGH ST7 TIMER PWM FUNCTION
AN 979	DRIVING AN ANALOG KEYBOARD WITH THE ST7 ADC
AN 980	ST7 KEYPAD DECODING TECHNIQUES, IMPLEMENTING WAKE-UP ON KEYSTROKE
AN1017	USING THE ST7 UNIVERSAL SERIAL BUS MICROCONTROLLER
AN1041	USING ST7 PWM SIGNAL TO GENERATE ANALOG OUTPUT (SINUSOÏD)
AN1042	ST7 ROUTINE FOR I²C SLAVE MODE MANAGEMENT
AN1044	MULTIPLE INTERRUPT SOURCES MANAGEMENT FOR ST7 MCUS
AN1045	ST7 S/W IMPLEMENTATION OF I²C BUS MASTER
AN1046	UART EMULATION SOFTWARE
AN1047	MANAGING RECEPTION ERRORS WITH THE ST7 SCI PERIPHERALS
AN1048	ST7 SOFTWARE LCD DRIVER
AN1078	PWM DUTY CYCLE SWITCH IMPLEMENTING TRUE 0% & 100% DUTY CYCLE
AN1082	DESCRIPTION OF THE ST72141 MOTOR CONTROL PERIPHERALS REGISTERS
AN1083	ST72141 BLDC MOTOR CONTROL SOFTWARE AND FLOWCHART EXAMPLE
AN1105	ST7 PCAN PERIPHERAL DRIVER
AN1129	PWM MANAGEMENT FOR BLDC MOTOR DRIVES USING THE ST72141
AN1130	AN INTRODUCTION TO SENSORLESS BRUSHLESS DC MOTOR DRIVE APPLICATIONS WITH THE ST72141
AN1148	USING THE ST7263 FOR DESIGNING A USB MOUSE
AN1149	HANDLING SUSPEND MODE ON A USB MOUSE
AN1180	USING THE ST7263 KIT TO IMPLEMENT A USB GAME PAD
AN1276	BLDC MOTOR START ROUTINE FOR THE ST72141 MICROCONTROLLER
AN1321	USING THE ST72141 MOTOR CONTROL MCU IN SENSOR MODE
AN1325	USING THE ST7 USB LOW-SPEED FIRMWARE V4.X
AN1445	EMULATED 16-BIT SLAVE SPI
AN1475	DEVELOPING AN ST7265X MASS STORAGE APPLICATION
AN1504	STARTING A PWM SIGNAL DIRECTLY AT HIGH LEVEL USING THE ST7 16-BIT TIMER
AN1602	16-BIT TIMING OPERATIONS USING ST7262 OR ST7263B ST7 USB MCUS
AN1633	DEVICE FIRMWARE UPGRADE (DFU) IMPLEMENTATION IN ST7 NON-USB APPLICATIONS
AN1712	GENERATING A HIGH RESOLUTION SINEWAVE USING ST7 PWMART
AN1713	SMBUS SLAVE DRIVER FOR ST7 I2C PERIPHERALS
AN1753	SOFTWARE UART USING 12-BIT ART

Table 31. ST7 Application Notes

IDENTIFICATION	DESCRIPTION
AN1947	ST7MC PMAC SINE WAVE MOTOR CONTROL SOFTWARE LIBRARY
GENERAL PURPOSE	
AN1476	LOW COST POWER SUPPLY FOR HOME APPLIANCES
AN1526	ST7FLITE0 QUICK REFERENCE NOTE
AN1709	EMC DESIGN FOR ST MICROCONTROLLERS
AN1752	ST72324 QUICK REFERENCE NOTE
PRODUCT EVALUATION	
AN 910	PERFORMANCE BENCHMARKING
AN 990	ST7 BENEFITS VS INDUSTRY STANDARD
AN1077	OVERVIEW OF ENHANCED CAN CONTROLLERS FOR ST7 AND ST9 MCUS
AN1086	U435 CAN-DO SOLUTIONS FOR CAR MULTIPLEXING
AN1103	IMPROVED B-EMF DETECTION FOR LOW SPEED, LOW VOLTAGE WITH ST72141
AN1150	BENCHMARK ST72 VS PC16
AN1151	PERFORMANCE COMPARISON BETWEEN ST72254 & PC16F876
AN1278	LIN (LOCAL INTERCONNECT NETWORK) SOLUTIONS
PRODUCT MIGRATION	
AN1131	MIGRATING APPLICATIONS FROM ST72511/311/214/124 TO ST72521/321/324
AN1322	MIGRATING AN APPLICATION FROM ST7263 REV.B TO ST7263B
AN1365	GUIDELINES FOR MIGRATING ST72C254 APPLICATIONS TO ST72F264
AN1604	HOW TO USE ST7MDT1-TRAIN WITH ST72F264
AN2200	GUIDELINES FOR MIGRATING ST7LITE1X APPLICATIONS TO ST7FLITE1XB
PRODUCT OPTIMIZATION	
AN 982	USING ST7 WITH CERAMIC RESONATOR
AN1014	HOW TO MINIMIZE THE ST7 POWER CONSUMPTION
AN1015	SOFTWARE TECHNIQUES FOR IMPROVING MICROCONTROLLER EMC PERFORMANCE
AN1040	MONITORING THE VBUS SIGNAL FOR USB SELF-POWERED DEVICES
AN1070	ST7 CHECKSUM SELF-CHECKING CAPABILITY
AN1181	ELECTROSTATIC DISCHARGE SENSITIVE MEASUREMENT
AN1324	CALIBRATING THE RC OSCILLATOR OF THE ST7FLITE0 MCU USING THE MAINS
AN1502	EMULATED DATA EEPROM WITH ST7 HDFLASH MEMORY
AN1529	EXTENDING THE CURRENT & VOLTAGE CAPABILITY ON THE ST7265 VDDF SUPPLY
AN1530	ACCURATE TIMEBASE FOR LOW-COST ST7 APPLICATIONS WITH INTERNAL RC OSCILLATOR
AN1605	USING AN ACTIVE RC TO WAKEUP THE ST7LITE0 FROM POWER SAVING MODE
AN1636	UNDERSTANDING AND MINIMIZING ADC CONVERSION ERRORS
AN1828	PIR (PASSIVE INFRARED) DETECTOR USING THE ST7FLITE05/09/SUPERLITE
AN1946	SENSORLESS BLDC MOTOR CONTROL AND BEMF SAMPLING METHODS WITH ST7MC
AN1953	PFC FOR ST7MC STARTER KIT
AN1971	ST7LITE0 MICROCONTROLLED BALLAST
PROGRAMMING AND TOOLS	
AN 978	ST7 VISUAL DEVELOP SOFTWARE KEY DEBUGGING FEATURES
AN 983	KEY FEATURES OF THE COSMIC ST7 C-COMPILER PACKAGE
AN 985	EXECUTING CODE IN ST7 RAM
AN 986	USING THE INDIRECT ADDRESSING MODE WITH ST7
AN 987	ST7 SERIAL TEST CONTROLLER PROGRAMMING
AN 988	STARTING WITH ST7 ASSEMBLY TOOL CHAIN
AN 989	GETTING STARTED WITH THE ST7 HIWARE C TOOLCHAIN

Table 31. ST7 Application Notes

IDENTIFICATION	DESCRIPTION
AN1039	ST7 MATH UTILITY ROUTINES
AN1064	WRITING OPTIMIZED HIWARE C LANGUAGE FOR ST7
AN1071	HALF DUPLEX USB-TO-SERIAL BRIDGE USING THE ST72611 USB MICROCONTROLLER
AN1106	TRANSLATING ASSEMBLY CODE FROM HC05 TO ST7
AN1179	PROGRAMMING ST7 FLASH MICROCONTROLLERS IN REMOTE ISP MODE (IN-SITU PROGRAMMING)
AN1446	USING THE ST72521 EMULATOR TO DEBUG AN ST72324 TARGET APPLICATION
AN1477	EMULATED DATA EEPROM WITH XFLASH MEMORY
AN1478	PORTING AN ST7 PANTA PROJECT TO CODEWARRIOR IDE
AN1527	DEVELOPING A USB SMARTCARD READER WITH ST7SCR
AN1575	ON-BOARD PROGRAMMING METHODS FOR XFLASH AND HDFLASH ST7 MCUS
AN1576	IN-APPLICATION PROGRAMMING (IAP) DRIVERS FOR ST7 HDFLASH OR XFLASH MCUS
AN1577	DEVICE FIRMWARE UPGRADE (DFU) IMPLEMENTATION FOR ST7 USB APPLICATIONS
AN1601	SOFTWARE IMPLEMENTATION FOR ST7DALI-EVAL
AN1603	USING THE ST7 USB DEVICE FIRMWARE UPGRADE DEVELOPMENT KIT (DFU-DK)
AN1635	ST7 CUSTOMER ROM CODE RELEASE INFORMATION
AN1754	DATA LOGGING PROGRAM FOR TESTING ST7 APPLICATIONS VIA ICC
AN1796	FIELD UPDATES FOR FLASH BASED ST7 APPLICATIONS USING A PC COMM PORT
AN1900	HARDWARE IMPLEMENTATION FOR ST7DALI-EVAL
AN1904	ST7MC THREE-PHASE AC INDUCTION MOTOR CONTROL SOFTWARE LIBRARY
AN1905	ST7MC THREE-PHASE BLDC MOTOR CONTROL SOFTWARE LIBRARY
SYSTEM OPTIMIZATION	
AN1711	SOFTWARE TECHNIQUES FOR COMPENSATING ST7 ADC ERRORS
AN1827	IMPLEMENTATION OF SIGMA-DELTA ADC WITH ST7FLITE05/09
AN2009	PWM MANAGEMENT FOR 3-PHASE BLDC MOTOR DRIVES USING THE ST7FMC
AN2030	BACK EMF DETECTION DURING PWM ON TIME BY ST7MC

15 IMPORTANT NOTES

Refer to Table 32 which provides a list of the trace codes for each of the recent silicon revisions.

Silicon revisions are identifiable:

– on the device package, by the last letter of the Trace Code marked on the device package.

– on the box, by the last 3 digits of the Internal Sales Type printed in the box label.

See also Figure 92. on page 137

Table 32. Device Identification

Device Type (Silicon Rev.)	Trace Code marked on device/ Internal Sales Type on box label
Flash Devices (Rev G) (Latest Flash silicon)	“xxxxxxxxxG” / 72F62xxxxx$x4
Flash Devices (Rev X) (Previous Flash silicon)	“xxxxxxxxxX” / 72F62xxxxx$x8 “xxxxxxxxxX” / 72F62xxxxx$x9
ROM Devices (Rev Z)	“xxxxxxxxxZ” / 7262xxxxx$x2 “xxxxxxxxxZ” / 7262xxxxx$x3

15.1 A/ D CONVERTER ACCURACY FOR FIRST CONVERSION

Description

When the ADC is enabled after being powered down (for example when waking up from HALT, ACTIVE-HALT or setting the ADON bit in the ADCCSR register), the first conversion (8-bit or 10-bit) accuracy does not meet the accuracy specified in the datasheet.

Workaround

In order to have the accuracy specified in the datasheet, the first conversion after a ADC switch-on has to be ignored.

Note:

This limitation does not apply to Flash silicon rev. G devices (see Table 32).

15.2 A/D CONVERTER CONVERSION SPEED

Description

Following a change in the fabrication location, the typical ADC conversion speed value for Flash devices has improved from a previous value of 28µs to 4µs.

When migrating software from Rev X to Rev G devices (refer to Table 32) care should be taken when using ADC interrupts.

Workaround

Firstly, on Rev G devices, only the use of One-Shot conversion mode is recommended in connection with ADC interrupts.

In Continuous Conversion mode, to avoid getting trapped in a continuous interrupt, the ADC interrupt routine must always have sufficient time to execute completely before the next ADC conversion interrupt, especially if the ADC interrupt is disabled outside of this routine. With the shorter conversion speed value, the interrupt may not be serviced fast enough. For this reason, on Rev G devices, the Continuous Conversion mode is not recommended in connection with ADC interrupts.

Secondly, in the interests of keeping code portable between all Flash/ROM versions, using the ADC as a source of a delayed trigger event is not advised. However, in such a scenario, a delay loop should be inserted for Rev G Flash devices to ensure that the timing remains the same for any such ADC delayed trigger events.

15.3 SCI WRONG BREAK DURATION

Description

A single break character is sent by setting and resetting the SBK bit in the SCICR2 register. In some cases, the break character may have a longer duration than expected:

– 20 bits instead of 10 bits if M=0

– 22 bits instead of 11 bits if M=1.

In the same way, as long as the SBK bit is set, break characters are sent to the TDO pin. This may lead to generate one break more than expected.

This affects all silicon revisions.

Occurrence

The occurrence of the problem is random and proportional to the baudrate. With a transmit frequency of 19200 baud (f_{CPU}=8MHz and SCIBRR=0xC9), the wrong break duration occurrence is around 1%.

Workaround

If this wrong duration is not compliant with the communication protocol in the application, software can request that an Idle line be generated before the break character. In this case, the break duration is always correct assuming the application is not doing anything between the idle and the break. This can be ensured by temporarily disabling interrupts.

The exact sequence is:

– Disable interrupts

– Reset and Set TE (IDLE request)

– Set and Reset SBK (Break Request)

– Re-enable interrupts

15.4 UNEXPECTED RESET FETCH

If an interrupt request occurs while a "POP CC" instruction is executed, the interrupt controller does not recognise the source of the interrupt and, by default, passes the RESET vector address to the CPU.

This affects all silicon revisions.

Workaround

To solve this issue, a "POP CC" instruction must always be preceded by a "SIM" instruction.

15.5 HALT MODE POWER CONSUMPTION WITH ADC ON

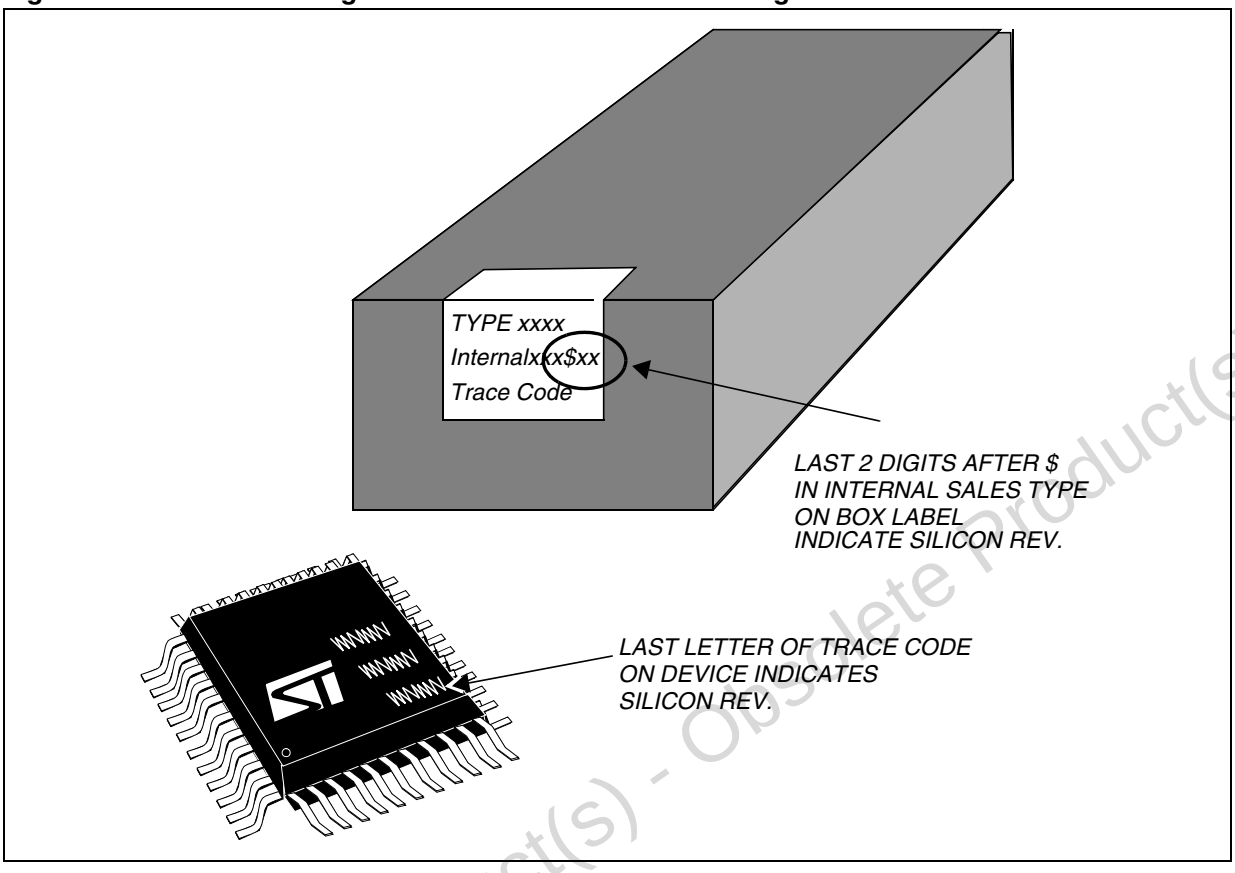
If the A/D converter is being used when Halt mode is entered, the power consumption in Halt Mode may exceed the maximum specified in the datasheet.

This affects all silicon revisions.

Workaround

Switch off the ADC by software (ADON=0) before executing a HALT instruction.

Figure 92. Revision Marking on Box Label and Device Marking



Note: Refer also to Table 32 on page 135 for additional revision identification notes

16 REVISION HISTORY

Description of the changes between the current release of the specification and the previous one.

Date	Revision	Description of Changes
23-Sep-2005	3.0	Clarification of Flash read-out protection in section 4.3.1 on page 14 Removed “optional” for V_{DD} in Figure 9 on page 15 Added one note in “Low Voltage Reset” on page 21 Added caution to “External clock and event detector mode” on page 47 Changed section 10.4.3.3 on page 59 Changed Table 18 on page 64 Changed section 10.5.4.3 on page 71 (noise error section) Changed “SCI Clock Tolerance” on page 74 Added “Noise Error Causes” on page 75 Added one row for Injected current on PA0 to PA7 pins in section 12.2.2 on page 102 Changed “EMC CHARACTERISTICS” on page 108 Changed figures and tables in “PACKAGE MECHANICAL DATA” on page 124 Changed description of FMP_R bit in section 14.1 on page 128 Added section 15.1 on page 135 Added section 15.3 on page 136 Added note in section 10.4.2 on page 56 Changed description of TC bit in section 10.5.7 on page 77 Modified maximum injected current values for PA0-PA6, PA7, section 12.2.2 on page 102 Reference made to the Flash Programming Reference Manual for Flash timing values section 12.6.2 on page 107 Updated option list Added figures and notes for RESET pin protection when LVD is enabled/disabled page 115 Added ECOPACK information in section 13 on page 124 Modified I_S value and corresponding note in section 12.8.1 on page 110
20-Mar-2006	4.0	All low voltage devices and characteristics removed Addition of Rs resistor in Figure 59. on page 106 Additional note added below Table 4 on page 20 Note added at end of section 10.7.6 on page 94 referring to Important notes Section 12.5.4 Crystal Oscillator Output Drive Level modified t_{CONV} values modified in section 12.12 on page 120 according to Flash/ROM silicon revision Important notes modified showing device identification Important note related to A/D converter conversion speed added, section 15.2 on page 135 Weak pull-up equivalent resistor values modified, section 12.9.1 on page 113 Graph in Figure 72 on page 114 updated Updated option list Notes updated for Section 12.9.1, Section 12.11.1, Section 12.11.2, Section 12.12.1 Figures modified for RESET pin protection when LVD is enabled/disabled page 115 All FASTROM options removed
11-Jun-2009	5	Updated part numbers on cover page. Replaced CCR by CC (Condition Code) register when the I bit is concerned. Added caution note in section 6.2 on page 21. Added note 1 below Section 12.2.3 Thermal Characteristics. Removed dynamic latchup in Section 12.7.3 Absolute Maximum Ratings (Electrical Sensitivity). Corrected CPHA value in Figure 79 on page 119. Updated $t_{V(MO)}$ and $t_{H(MO)}$ in Figure 80 on page 119, and in Section 12.11.2 SPI - Serial Peripheral Interface. Updated ECOPACK text in section 13 on page 124.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com