SIEMENS

Microcomputer Components
Technical Support Group Munich
HL DC AT

Errata Sheet

December 14, 1997 / Release 1.0

Device : **SAK-C167CR-4RM**

Stepping Code / Marking : **ES-AC**

The C167CR**-4RM** is the **32 Kbyte ROM** version of the C167CR, including an on-chip CAN module, 2 Kbyte XRAM module, and a PLL oscillator circuit.

This errata sheet describes the functional problems known in this step. Problem classification and numbering is performed relative to modules, where the C167 AC-step is the reference. Since most problems of earlier steps have already been fixed in this step of the C167CR, problem numbering is not necessarily consecutive.

The C167CR-4RM devices are mounted in a 144-pin Plastic Metric Quad Flat Pack (P-MQFP-144-1) package.

Note: devices which are marked as **ES-AC** are **engineering samples**. Specific test conditions/restrictions may apply to these engineering samples which may differ from the standard test conditions and specifications. These are described in a separate **Status Sheet**.

Changes from Errata Sheet **Rel. 1.0** for **C167CR-4RM** devices with stepping code/marking **ES-AA** to this Errata Sheet **Rel. 1.0** for **C167CR-4RM** devices with stepping code/marking **ES-AC:**

- Problems ADC.10, CPU.15 fixed
- Bidirectional Hardware Reset (RST.3)
- Execution of PWRDN Instruction while pin NMI# = high (PWRDN.1)
- Arithmetic Overflow by DIVLU instruction (CPU.17)
- Data read access with MOVB [Rn], mem instruction to internal ROM (CPU.16)
- Deviations from DC/AC Specification: Output Low Voltage Test Condition (DCVOL.1)

Functional Problems

The following malfunctions are known in this step:

RST.3: Bidirectional Hardware Reset

When the bidirectional reset feature is enabled (bit BDRSTEN/SYSCON.3 = 1), and a short hardware reset pulse (> 4 TCL) is applied to pin RSTIN#, the following problem may occur:

Pin RSTIN# may not be driven low by the internal circuitry for the duration of the internal reset sequence if the falling edge of the hardware reset pulse occurred during the second cycle of a 2-cycle instruction (e.g. CALL, RETI, TRAP). As a consequence, the level at pin RSTIN# may be pulled up to a high level through the internal pullup in case the external reset source is no longer driving the pin low.

Note:

The internal reset sequence is always completed correctly. Software and Watchdog Timer reset are not affected by this problem.

This problem will be fixed in the next step

PWRDN.1: Execution of PWRDN Instruction while pin NMI# = high

When instruction PWRDN is executed while pin NMI# is at a high level, power down mode should not be entered, and the PWRDN instruction should be ignored. However, under the conditions described below, the PWRDN instruction may not be ignored, and no further instructions are fetched from external memory, i.e. the CPU is in a quasi-idle state. This problem will only occur in the following situations:

a) the instructions following the PWRDN instruction are located in external memory, and a **muliplexed bus** configuration **with memory tristate waitstate** (bit MTTCx = 0) is used, or

b) the instruction preceeding the PWRDN instruction **writes** to external memory or an XPeripheral (XRAM, CAN), and the instructions following the PWRDN instruction are located in external memory. In this case, the problem will occur for any bus configuration.

Note: the on-chip peripherals are still working correctly, in particular the Watchdog Timer will reset the device upon an overflow. Interrupts and PEC transfers, however, can not be processed. In case NMI# is asserted low while the device is in this quasi-idle state, power down mode is entered.

Workaround:

Ensure that no instruction which writes to external memory or an XPeripheral preceeds the PWRDN instruction, otherwise insert e.g. a NOP instruction in front of PWRDN. When a muliplexed bus with memory tristate waitstate is used, the PWRDN instruction should be executed out of internal RAM or XRAM.

CPU.17: Arithmetic Overflow by DIVLU instruction

For specific combinations of the values of the dividend (MDH,MDL) and divisor (Rn), the Overflow (V) flag in the PSW may not be set for **unsigned divide** operations, although an overflow occured.

E.g.:

```
MDH  MDL     Rn        MDH  MDL
F0F0  0F0Fh : F0F0h  = FFFF FFFFh, but no Overflow indicated !
                       (result with 32-bit precision: 1 0000h)
```

The same malfunction appears for the following combinations:

```
n0n0 0n0n : n0n0
n00n 0nn0 : n00n
n000 000n : n000
n0nn 0nnn : n0nn    where n means any Hex Digit between 8 ...  F
```

i.e. all operand combinations where at least the most significat bit of the dividend (MDH) and the divisor (Rn) is set.

In the cases where an overflow occurred after DIVLU, but the V flag is not set, the result in MDL is equal to FFFFh.

Workaround:

Skip execution of DIVLU in case an overflow would occur, and explicitly set V = 1.

```
E.g.:               CMP Rn, MDH
                    JMPR cc_ugt, NoOverflow ; no overflow if Rn > MDH
                    BSET V                  ; set V = 1 if overflow would occur
                    JMPR cc_uc, NoDivide    ; and skip DIVLU
    NoOverflow:     DIVLU Rn
    NoDivide: ...               ; next instruction, may evaluate correct V flag
```

Note:

- the KEIL C compiler, run time libraries and operating system RTX166 do not generate or use instruction sequences where the V flag in the PSW is tested after a DIVLU instruction.

- with the TASKING C166 compiler, for the following intrinsic functions code is generated which uses the overflow flag for minimizing or maximizing the function result after a division with a DIVLU:

```
_div_u32u16_u16()
_div_s32u16_s16()
_div_s32u16_s32()
```

Consequently, an incorrect overflow flag (when clear instead of set) might affect the result of one of the above intrinsic functions but only in a situation where no correct result could be calculated anyway. These intrinsics first appeared in version 5.1r1 of the toolchain.

Libraries: not affected

CPU.16: Data read access with MOVB [Rn], mem instruction to internal ROM

When the *MOVB [Rn], mem* instruction (opcode 0A4h) is executed, where

1. *mem* specifies a direct 16-bit byte operand address in the internal ROM/Flash memory,

AND

2. *[Rn]* points to an **even** byte address, while the contents of the word which includes the byte addressed by *mem* is **odd,**

   OR

   [Rn] points to an **odd** byte address, while the contents of the word which includes the byte addressed by *mem* is **even**

the following problem occurs:

a) when *[Rn]* points to **external** memory or to the **X-Peripheral** (XRAM, CAN) address space, the data value which is written back is always 00h

b) when *[Rn]* points to the **internal** RAM or SFR/ESFR address space,

- the (correct) data value *[mem]* is written to *[Rn]***+1**, i.e. to the **odd** byte address of the selected word in case *[Rn]* points to an **even** byte address,
- the (correct) data value *[mem]* is written to *[Rn]***-1**, i.e. to the **even** byte address of the selected word in case *[Rn]* points to an **odd** byte address.

Workaround:

When *mem* is an address in internal ROM, substitute instruction

MOVB [Rn], mem e.g. by *MOV Rm, #mem*
MOVB[Rn], [Rm]

Note: the Keil C166 Compiler V3.10 has been extended by the directive FIXROM which avoids accesses to 'const' objectes via the instruction MOVB [Rn], mem.

Functional Problem	Short Description	Remarks
RST.3	Bidirectional Hardware Reset	**step AC only**
PWRDN.1	Execution of PWRDN Instruction while pin NMI# = high	
CPU.17	Arithmetic Overflow by DIVL Instructions	
CPU.16	Data read access with MOVB [Rn], mem instruction to internal ROM	
PLL.1	PLL Unlock Behaviour	**fixed in step AB-ES**
CPU.15	Data Access to internal ROM Variables by Arithmetic/Logical Instructions	**fixed in step AC**
ADC.10	Start of Standard Conversion at end of Injected Conversion	**fixed in step AC**

Table 1: Functional Problems of the C167CR-4RM

Specific Problems with X-Peripherals (XPERs)

The following problems with the interface to XPERs, the CAN module, and the XRAM module are currently known:

X9: Read Access to XPERs in Visible Mode

The data of a read access to an XBUS-Peripheral (XRAM, CAN) in Visible Mode is not driven to the external bus. PORT0 is tristated during such read accesses.

Functional Problem	Short Description	fixed in Step
X9	Read Access to XPERs in Visible Mode	

Table 2: Functional Problems with XPERs on the C167CR-4RM

Deviations from DC/AC Specification

The following table lists the deviations of the DC/AC characteristics from the specification in the C167CR-4RM Data Sheet 7.97.

Problem short name	Parameter	Symbol	Limit Values		Unit	Test Condition
			min.	max.		
DCVOL.1	Output low voltage (Port0/1/4, ALE, RD#, WR#, WRH#/BHE, CLKOUT, RSTOUT#)	V_{OL}	**-**	0.45	V	I_{OL} = **2.0 mA** instead of 2.4 mA
DCAH.1	ALE active current	I_{ALEH}	**1000** instead of 500	-	µA	V_{OUT} = 2.4 V
DCRL.1	RD#/WR# active current	I_{RWL}	**-600** instead of -500	-	µA	$V_{OUT} = V_{OLmax}$
DCP6L.1	Port 6 active current	I_{P6L}	**-600** instead of -500	-	µA	$V_{OUT} = V_{OLmax}$
DCHYS.1	Input Hysteresis (Special Threshold)	HYS	**300** instead of 400	-	mV	-

Table 3: Deviations from DC/AC Specification of the C167CR-4RM

Notes:

1) Pin **READY**# has an internal pullup (all C167xx derivatives). This will be documented in the next revision of the Data Sheet.

2) Timing **t28**: Parameter description and test changed from ’Address hold after RD#/WR#’ to ’Address hold after WR#’. It is guaranteed by design that read data are internally latched by the controller before the address changes.

3) During **reset**, the **internal pullups on P6.[4:0]** are active, independent whether the respective pins are used for CS# function after reset or not.

In addition to the description in the C167 Derivatives User's Manual V2.0, the following feature enhancements have been implemented in the C167CR-4RM:

Incremental position sensor interface

For each of the GPT1 timers T2, T3, T4 of the GPT1 unit, an additional operating mode has been implemented which allows to interface to incremental position sensors (A, B, Top0). This mode is selected for a timer Tx via TxM = 110b in register TxCON, x = (2, 3, 4). Optionally, the contents of T5 may be captured into register CAPREL upon an event on T3. This feature is selected via bit CT3 = 1 in register T5CON.10

Compatibility with previous versions:

In previous versions (e.g. C167CR-LM), both of the settings (TxM = 110b, T5CON.10 = 1) were reserved and should not be used. Therefore, systems designed for previous versions will also work without problems with the C167CR-4RM.

Oscillator Watchdog

The C167CR-4RM provides an Oscillator Watchdog (OWD) which monitors the clock at XTAL1 in direct drive mode. In case of clock failure, the PLL Unlock/OWD Interrupt Request Flag (XP3IR) is set and the internal CPU clock is supplied with the PLL basic frequency. This feature can be disabled by a low level on pin Vpp/OWE. See also C167CR-4RM Data Sheet 7.97.

Bidirectional Reset

The C167CR-4RM allows to indicate an internal watchdog timer or software reset on the RSTIN# pin which will be driven low for the duration of the internal reset sequence. This option is selectable by software via bit BDRSTEN/SYSCON.3. After reset, the bidirectional reset option is disabled (BDRSTEN/SYSCON.3 = 0). See also C167CR-4RM Data Sheet 7.97. Beginning with the **AC-step** of the **C167CR-4RM**, RSTIN# will also be driven low for the duration of the internal reset sequence when this reset was initiated by an **external HW reset** signal on pin RSTIN#.

Please note also the following functional difference to the C167CR-LM BA-step:

XBUS Peripheral Enable Bit XPEN/SYSCON.2

In the C167CR-4RM, bit SYSCON.2 is a general XBUS Peripheral Enable bit, i.e. it controls both the XRAM **and the CAN module**.

Compatibility with previous versions:

When bit SYSCON.2 = 0 (default after reset) in the C167CR-4RM, and an access to an address in the range EF00h ... EFFFh is made, either an external bus access is performed (if an external bus is enabled), or the Illegal Bus Trap is entered. In previous versions (e.g. C167CR-LM), the CAN module was accessed in this case.

Systems where bit SYSCON.2 was set to '1' before an access to the CAN module in the address range EF00h ... EFFFh was made will also work without problems with the C167CR-4RM.