Microcontrollers

Errata Sheet

October 16, 2001 / Release 1.5

Device:	**SAK-C167CS-LM** **SAK-C167CS-4RM**
Stepping Code / Marking:	**ES-AB, AB**
Package:	**P-MQFP-144-1**

This Errata Sheet describes the deviations from the current user documentation. The classification and numbering system is module oriented in a continual ascending sequence over several derivatives, as well already solved deviations are included. So gaps inside this enumeration could occur.

The current documentation is:
Data Sheet: C167CS Data Sheet V2.0 2000-06
User's Manual: C167CS User's Manual V2.0 2000-07
Instruction Set Manual V2.0, 2001-03

Note: ***Devices marked with EES or ES are engineering samples which may not be completely tested in all functional and electrical characteristics, therefore they should be used for evaluation only.***

The specific test conditions for EES and ES are documented in a separate Status Sheet.

Change summary to Errata Sheet Rel.1.4 for C167CS-4RM/LM devices with stepping code/marking ES-AB, AB:

- reference to Instruction Set Manual V2.0, 2001-03
- Contents of Message Objects and Mask of Last Message Registers after Reset (CAN.9)
- BFLDL/H Instruction after Write Operation to internal IRAM (CPU.21): complete problem description included
- Application Hint ' PLL lock after temporary clock failure' added

Functional Problems:

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 **multiplexed bus** configuration **with memory tristate waitstate** (bit MTTCx = 0) is used, or
b) the instruction preceding 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 precedes the PWRDN instruction, otherwise insert e.g. a NOP instruction in front of PWRDN. When a multiplexed bus with memory tristate waitstate is used, the PWRDN instruction should be executed out of internal RAM or XRAM.

ADC.11: Modifications of ADM field while bit ADST = 0

The A/D converter may unintentionally start one auto scan single conversion sequence when the following sequence of conditions is true:

(1) the A/D converter has finished a fixed channel single conversion of an analog channel n > 0 (i.e. contents of ADCON.ADCH = n during this conversion)
(2) the A/D converter is idle (i.e. ADBSY = 0)
(3) then the conversion mode in the ADC Mode Selection field ADM is changed to Auto Scan Single (ADM = 10b) or Continuous (ADM = 11b) mode without setting bit ADST = 1 with the same instruction

Under these conditions, the A/D converter will unintentionally start one auto scan single conversion sequence, beginning with channel n-1, down to channel number 0.

In case the channel number ADCH has been changed before or with the same instruction which selected the auto scan mode, this channel number has no effect on the unintended auto scan sequence (i.e. it is not used in this auto scan sequence).

Note:
When a conversion is already in progress, and then the configuration in register ADCON is changed,

- the new conversion mode in ADM is evaluated after the current conversion
- the new channel number in ADCH and new status of bit ADST are evaluated after the current conversion when a conversion in fixed channel conversion mode is in progress, and after the current conversion sequence (i.e. after conversion of channel 0) when a conversion in an auto scan mode is in progress.

In this case, it is a specified operational behaviour that channels n-1 .. 0 are converted when ADM is changed to an auto scan mode while a fixed channel conversion of channel n is in progress (see e.g. C167CS User's Manual, V2.0, p18-6)

Workaround:

When an auto scan conversion is to be performed, always start the A/D converter with the same instruction which sets the configuration in register ADCON.

CAPCOM.4: SW Access to P1H Overwrites CAPCOM HW Settings

HW settings on P1H.7...4 by CAPCOM compare output functions (CC24 ... CC27) can be overwritten by SW accesses to P1H.7...0 on the same port.

Read modify write operations like BSET, BFLDx, OR, ... read the input or output latches respectively, modify the affected bits and write back the result to the output latches of the whole port (P1H7...0).

In case a compare event has occurred after the read phase, but before the write-back phase of such an instruction, the output signal change of the compare event is lost or only a short pulse (≥1 TCL) may appear. The bit protection mechanism to avoid these effects is out of function on P1H.

Workarounds:

1. Avoid the combination of HW and SW write accesses to P1H. HW access only to CC24 ... CC27 or SW access only to P1H.7 ... 0 works properly.

2. Use „interrupt only“ compare modes (CCMODx = 100 or 110) and modify the port pin in the interrupt service routine by software to avoid the combination of HW and SW accesses.

RST.8: Clock failure detection during external Reset

A missing clock at pin XTAL1 during an external reset cannot be recognized via bit PLLIR (PLL/OWD interrupt request flag) in register ISNC (Interrupt Subnode Control) because bit PLLIR is cleared during reset. When the clock at pin XTAL 1 is missing the internal CPU clock is the PLL base frequency (2...5 MHz).

Workaround:

(1) Clock options at PORT0 (P0H.7 ... 5) are set to Direct Drive or Prescaler mode:

Use the Real Time Clock with main oscillator (default after Power-on reset) as input clock source. Check bit RTCIR (T14IR) of the Real Time Clock instead of bit PLLIR. If the RTC interrupt request does not occur in the expected time frame then the main oscillator does not oscillate.

Instead of an interrupt controlled verification of the RTC activity also polling of register T14 for at least 256 XTAL1 cycles can be used to check whether the main oscillator is running.

(2) Clock options at PORT0 (P0H.7 ... 5) are set to PLL mode:

Bit CLKLOCK in SYSCON2 can be tested instead of bit PLLIR. CLKLOCK = 0 while the PLL is unlocked. The workaround described in case (1) can also be used instead of the CLKLOCK bit.

RST.9.2: Software/Watchdog/Short HW Reset during Slow Down operation with PLL off

When a watchdog timer or software reset or short hardware reset occurs during slow down mode where the PLL has been switched off (CLKCON = 10 in register SYSCON2), the internal reset condition is extended until the next long HW reset.

After the reset, if the PLL has not locked, flag PLLIR in register ISNC is **not set, but** the device continues operation on the PLL base frequency (2 .. 5 MHz).

This problem only occurs when the PLL is used to generate the internal clock during normal operation, and it will not occur in direct drive or prescaler mode.

Workarounds:

Enable bi-directional reset mode (bit BDRSTEN = 1 in register SYSCON). The external reset circuit should prolong this reset until the PLL is locked again (approx. 1 ms).

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.

Note that in Visible Mode PORT1 will drive the address for an access to an XBUS-Peripheral, even when only a multiplexed external bus is enabled.

X17: XBUS Access after External Access with EWEN = 1 and/or BSWC = 1

1. When the last external bus access which precedes an internal XBUS-Peripheral access is performed with bit **BUSCONx.11/BSWCx = 1 (BUSCON Switch Tristate Waitstate)**, the XBUS cycle will be performed without BUSCON Switch Tristate Waitstate, and the BUSCON Switch Tristate Waitstate will be inserted at the beginning of the next external bus cycle.

   Workaround: the data of an internal XBUS write cycle will be driven on Port 0 starting 2 TCL after the rising edge of ALE. In case the tdf (output disable time) specification of the external device does not meet the following requirements

   tdf < 4 TCL – 15 + tf [ns] for a multiplexed bus cycle via BUSCONx, or
   tdf < 2 TCL – 15 + tf [ns] for a non-multiplexed bus cycle via BUSCONx,

   then use a standard Memory Tristate Time Waitstate instead of a BUSCON Switch Tristate Waitstate, i.e. set bit BUSCONx.5/MTTCx = 0.

2. When the last external bus access which precedes an internal XBUS-Peripheral access is performed with bit **BUSCONx.8/EWENx = 1 (Early Write)**, the XBUS cycle will also be performed with 'Early Write'.

   Workaround: not required, internal XBUS-Peripherals tolerate write accesses with Early Write.

TCOMP.3: Pad Driver Temperature Compensation disabled

To avoid a malfunction and additional current consumption, the on chip pad driver temperature compensation is disabled by default after reset. For this reason, the functionality of some bits has been inverted:

- TCS = 0 (default after reset) means that the port logic is controlled by bit field TCC
- TCC = 00 (default after reset) means that the pad drivers operate with maximum driver strength

Workaround:

Not necessary if normal (strong) pad drivers shall be used, otherwise bit field TCC may be modified. Bit TCS should never be set to '1'.

POWER.15: Sleep Mode not possible in Prescaler Mode

In prescaler mode ($f_{CPU} = f_{OSC}/2$) instead of sleep the device enters an undefined state and no wake up is possible except HW reset.

Workaround:

Avoid sleep mode with prescaler mode (use deep idle instead: all peripherals off and slow down divider/32 and idle mode)

POWER.16: SDD and Sleep Mode cannot be terminated

When the power off/on recovery time is too short there can be a problem with internal clock lock detection. Due to this the Slow Down Divider Mode and the Sleep Mode (for derivatives with sleep mode implemented only) cannot be terminated (as they wait for the clock lock condition).

Note: In this case the bit CLKLOCK in register SYSCON2 always remains cleared.

This problem can occur when the power off/on recovery time is below 4 seconds (beginning from ≤ 0.3V at all pins) as an empirical value.

Workaround:

Avoid power off/on recovery time < 4 seconds.

SYSCON1.1: Reset Value in Register SYSCON1

The reset value in register SYSCON1 is 0FF00h instead of 0000h. This has no functional impact, since currently only for the two least significant bits (SLEEPCON) a function is defined.

BUS.19: Unlatched Chip Selects at Entry into Hold Mode

Unlike in standard (latched) configuration, the chip select lines in unlatched configuration (SYSCON.CSCFG = 1) are not driven high for 1 TCL after HLDA# is driven low, but start to float when HLDA# is driven low.

CPU.21 BFLDL/BFLDH Instructions after Write Operations to internal IRAM

The result of a BFLDL/BFLDH (=BFLDx) instruction may be incorrect if the following conditions are true at the same time:

(1) the previous 'instruction' is a PEC transfer which writes to IRAM, or any instruction with result write back to IRAM (addresses 0F200h..0FDFFh for 3 Kbyte module, 0F600h..0FDFFh for 2 Kbyte module, or 0FA00h..0FDFFh for 1 Kbyte module). For further restrictions on the destination address see case (a) or case (b) below.

(2) the BFLDx instruction immediately follows the previous instruction or PEC transfer within the instruction pipeline ('back-to-back' execution), i.e. decode phase of BFLDx and execute phase of the previous instruction or PEC transfer coincide. This situation typically occurs during program execution from internal program memory (ROM/OTP/Flash), or when the instruction queue is full during program execution from external memory

(3) the 3rd byte of BFLDx (= ***#mask8*** field of BFLD**L** or ***#data8*** field of BFLD**H**) and the destination address of the previous instruction or PEC transfer match in the following way:

(a) value of #mask8 of BFLDL or #data8 of BFLDH = 0F**y**h (**y** = 0..Fh),
and the previous instruction or PEC writes to (the low and/or high byte of) GPR R**y** or the memory address of R**y** (determined by the context pointer CP) via any addressing mode.

(b) value of #mask8 of BFLDL or #data8 of BFLDH = **00h..0EFh**,
and the lower byte $\mathbf{v_L}$ of the **contents v** of the IRAM location or (E)SFR or GPR which is read by BFLDx is **00h** £ $\mathbf{v_L}$ £ **7Fh**
and the previous instruction or PEC transfer writes to the (low and/or high byte of) the specific bit-addressable IRAM location **0FD00h + 2** $\mathbf{v_L}$ (i.e. the 8-bit offset address of the location in the bit-addressable IRAM area (0FD00h..0FDFFh) equals $\mathbf{v_L}$).

When the problem occurs, the actual result (all 16 bits) of the BFLDx instruction is bitwise ORed with the (byte or word) result of the previous instruction or PEC transfer.

Notes:

Write operations in the sense of the problem description include implicit write accesses caused by
- auto-increment operations of the PEC source or destination pointers (which are located on 0FCE0h..0FCFEh in IRAM)
- post-increment/pre-decrement operations on GPRs (addressing modes with [R+] or [-R])
- write operations on the system stack (which is located in IRAM).

In case **PEC write operations** to IRAM locations which match the above criteria (bit-addressable or active register bank area, PEC pointers not overlapping with register bank area) can be **excluded**, the problem will **not** occur when the instruction preceding BFLDx in the dynamic flow of the program is one of the following instructions (which do not write to IRAM):

NOP
ATOMIC, EXTx
CALLA/CALLI/JBC/JNBS when branch condition = false
JMPx, JB, JNB
RETx (except RETP)
CMP(B) (except addressing mode with [Rwi+]), BCMP
MULx, DIVx
IDLE, PWRDN, DISWDT, SRVWDT, EINIT, SRST

For implicit IRAM write operations caused by **auto-increment operations of the PEC source or destination pointers**, the problem can only occur if the value of #mask8 of BFLDL or #data8 of BFLDH = 0F**y**h (**y** = 0..Fh), and the range which is covered by the context pointer CP (partially or completely) overlaps the PEC source and destination pointer area (0FCE0h..0FCFEh), and the address of the source or destination pointer which is auto-incremented after the PEC transfer is equal to the address of GPR R**y** (included in case 3a).

For **system stack write operations**, the problem can only occur if the system stack is located in the bit-addressable portion of IRAM (0FD00h..0FDFFh), or if the system stack can overlap the register bank area (i.e. the register bank area is located below the system stack, and the distance between the contents of the context pointer CP and the stack pointer SP is ≤ 20h).

Workaround 1:

When a critical instruction combination or PEC transfer to IRAM can occur, then substitute the BFLDx instruction by

(a) an equivalent sequence of single bit instructions. This sequence may be included in an uninteruptable ATOMIC or EXTEND sequence to ensure completion after a defined time.

(b) an equivalent byte- or word MOV or logical instruction.

Note that byte operations to SFRs always clear the non-addressed complementary byte.
Note that protected bits in SFRs are overwritten by MOV or logical instructions.

Workaround 2:

When a critical instruction combination occurs, and **PEC write operations** to IRAM locations which match the above criteria (bit-addressable or active register bank area) **can be excluded**, then rearrange the BFLDx instruction within the instruction environment such that a non-critical instruction sequence is generated.

Workaround 3:

When a critical instruction combination or PEC transfer to IRAM can occur, then

- replace the BFLDx instruction by the instruction sequence
  ATOMIC #1
  BFLDx

  This means e.g. when BFLDx was a branch target before, ATOMIC # 1 is now the new branch target.

In case the BFLDx instruction is included at position ***n*** in an ATOMIC or EXTEND sequence with range operator #**m** (n, m = 2..4, n ≤ m), then

- insert (repeat) the corresponding ATOMIC or EXTEND instruction at position ***n*** with range operator #**z** where **z** = (***m*** - ***n***) + 1

	Range of original ATOMIC/EXTEND statement			
Position of BFLDx within ATOMIC/ EXT.. sequence	1	2	3	4
1	no problem / no workaround	no problem / no workaround	no problem / no workaround	no problem / no workaround
2	- -	z = 1	z = 2	z = 3
3	- -	- -	z = 1	z = 2
4	- -	- -	- -	z = 1

- - : case can not occur

Tool Support for Problem CPU.21

The **Keil** C166 Compiler V3.xx generates BFLD instructions only in the following cases:
- when using the _bfld_ intrinsic function.
- at the beginning of interrupt service routines, when using #pragma disable
- at the end of interrupt service routines, when using the chip bypass directive FIX166.

The C166 Compiler V4.xx uses the BFLD instruction to optimize bit-field struct accesses. Release C166 V4.10 offers a new directive called FIXBFLD that inserts an ATOMIC #1 instruction before every BFLD instruction that is not enclosed in an EXTR sequence. Detailed information can be found in the C166\HLP\RELEASE.TXT of C166 Version 4.10.

The C166 Run-Time Library for C166 V3.xx and V4.xx uses BFLD instructions only in the START167.A66 file. This part of the code should be not affected by the CPU.21 problem but should be checked by the software designer.

The RTX166 Full Real-Time Operating system (any version) does not use BFLD instructions.
For RTX166 Tiny, you should rebuild the RTX166 Tiny library with the SET FIXBFLD = 1 directive. This directive is enabled in the assembler source file RTX166T.A66. After change of this setting rebuild the RTX166 Tiny library that you are using in your application.

The **Tasking** support organization provides a v7.0r1 A166 Assembler (build 177) including a check for problem CPU.21 with optional pec/no_pec feature. This assembler version can also be used to check code which was generated with previous versions of the Tasking tool chain. A v7.0r1 C166 Compiler (build 368) offering a workaround for problem CPU.21 is also available from Tasking.

The scan tool **aiScan21** analyzes files in hex format plus user-supplied additional information (locator map file, configuration file), checks whether they may be affected by problem CPU.21, and produces diagnostic information about potentially critical instruction sequences. This tool is included in AP1628 'Scanning for Problem CPU.21' on
http://www.infineon.com/cgi/ecrm.dll/ecrm/scripts/prod_cat.jsp?oid=-8137

CAN.9: Contents of Message Objects and Mask of Last Message Registers after Reset

After any reset, the contents of the CAN Message Objects 1..15 (MCR, UAR, LAR, MCFG, Data[0:7]) and the Mask of Last Message Registers (LMLM, UMLM) may be undefined instead of unchanged (reset value 'X' instead of 'U').

This problem depends on temperature and the length of the reset, and differs from device to device. The problem is more likely (but not restricted) to occur at high temperature and for long hardware resets (> 100 ms).

Workaround:

Re-initialize the CAN module after each reset.

CAN.7 Unexpected remote frame transmission

Symptom

The on-chip CAN module may send an unexpected remote frame with the identifier=0, when a pending transmit request of a message object is disabled by software.

Detailed Description

There are three possibilities to disable a pending transmit request of a message object (n=1..14):

- Set CPUUPDn element
- Reset TXRQn element
- Reset MSGVALn element

Either of these actions will prevent further transmissions of message object n.

The symptom described above occurs when the CPU accesses CPUUPD, TXRQ or MSGVAL, while the pending transmit request of the corresponding message object is transferred to the CAN state machine (just before start of frame transmission). At this particular time the transmit request is transferred to the CAN state machine before the CPU prevents transmission. In this case the transmit request is still accepted from the CAN state machine. However the transfer of the identifier, the data length code and the data of the corresponding message object is prevented. Then the pre-charge values of the internal "hidden buffer" are transmitted instead, this causes a remote frame transmission with identifier=0 (11 bit) and data length code=0.
This behavior occurs only when the transmit request of message object n is pending and the transmit requests of other message objects are **not** active (single transmit request).
If this remote frame loses arbitration (to a data frame with identifier=0) or if it is disturbed by an error frame, it is **not** retransmitted.

Effects to other CAN nodes in the network

The effect leads to delays of other pending messages in the CAN network due to the high priority of the Remote Frame. Furthermore the unexpected remote frame can trigger other data frames depending on the CAN node's configuration.

Workarounds

1. The behavior can be avoided if a message object is not updated by software when a transmission of the corresponding message object is pending (TXRQ element is set) **and** the CAN module is active (INIT = 0). If a re-transmission of a message (e.g. after lost arbitration or after the occurrence of an error frame) needs to be cancelled, the TXRQ element should be cleared by software as soon as NEWDAT is reset from the CAN module.
2. The nodes in the CAN system ignore the remote frame with the identifier=0 and **no** data frame is triggered by this remote frame.

Deviations from Electrical- and Timing Specification:

The following table and sections describe the deviations of the DC/AC characteristics from the specification in the C167CS Data Sheet 2000-06:

Problem short name	Parameter	Symbol	Limit Values		Unit	Test Condition
			min.	max.		
DC.VILS.2	Input Low Voltage (Special Threshold)	V_{ILS}	-0.5	**1.8** instead of 2.0	V	
DC.HYS.2	Input Hysteresis (Special Threshold)	HYS	**200** instead of 400	-	mV	

DC.IID_PD.1: Idle and Power Down Mode Supply Currents I_{IDX}, I_{IDO}, I_{PDR}: not tested

The Idle Mode Supply Currents

I_{IDX} (all peripherals active) and I_{IDO} (all peripherals and PLL off, SDD = 32)

and the Power Down Mode Supply Current

I_{PDR} (RTC running)

are not tested.

PLL.3.4: Increased PLL Jitter caused by external Access

Problem description:

In systems where the PLL is used for generation of the CPU clock frequency, the PLL jitter can increase under certain circumstances and exceed the specified value.
The value of the additional jitter is not a fixed one but it depends on the kind of activity on the external bus or output pins. The additional jitter increases with the number of output/bus pins which are switched at the same time to a new voltage level because the problem is caused by noise on the on-chip power supply.
The capacitive load of the used pins has also an influence to the additional jitter. A high capacitive load can increase the additional jitter.

Effects to the system:

All PLL factors are affected (PLL factor 1.5 / 2 / 2.5 / 3 / 4 / 5). The PLL jitter increases when access to the external bus or output pins is performed. This phenomenon has no influence to direct drive-, prescaler-, and SDD mode.
The problem does not affect the functionality of the CPU and the on-chip peripherals.
The additional jitter has the maximum effect if only one TCL is considered (period jitter). This can have an influence to the bus timings with one TCL.
The additional jitter decreases with the number of consecutive TCLs (accumulated jitter). The accumulated jitter has an effect on timings with more than one TCL (certain bus timings, CAN bus timing, serial interface).

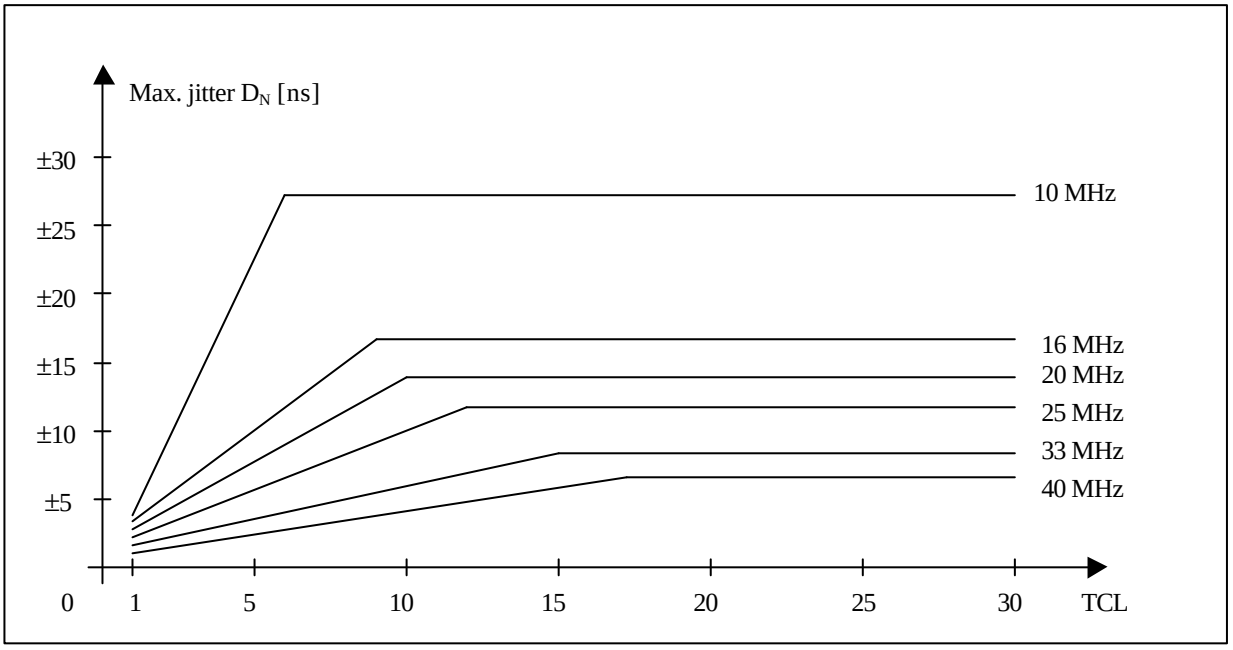
New value of the PLL jitter

For an accumulated period of N*TCL the new maximum jitter D_N[ns], which exceeds the specified value in the Data Sheet, is computed using the formula:

$$D_N[ns] = \pm((260 / f_{CPU} + 100 / f_{CPU}^2) * (N / Tm) + 0.2)$$

Were N = number of consecutive TCL, f_{CPU} = CPU frequency in MHz and $Tm = f_{CPU} * 0.35 + 2.5$.
This approximated formula is valid for a CPU frequency 10 MHz ≤ f_{CPU} ≤ 40 MHz with the conditions:

$Tm = f_{CPU} * 0.35 + 2.5$ for N £ Tm
$Tm = N$ for N > Tm



Workaround:

In the case that the additional jitter causes problems in the system which cannot be relaxed with software adapted timing parameters the following workarounds are possible:

- If the clock generation is done with the on-chip oscillator then direct drive- or prescaler mode can be used.
- If the maximum frequency of the on-chip oscillator in not sufficient then an external clock generator can be used.

A/D Converter Characteristics:

ADCC.2.5: ADC Overload Current

During exceptional conditions in the application system an overload current I_{OV} can occur on the analog inputs of the A/D converter when $V_{AIN} > V_{DD}$ or $V_{AIN} < V_{SS}$. For this case, the following conditions are specified in the Data Sheet:

I_{OVmax} = | ±5 mA |

The specified total unadjusted error TUE_{max} = | ±2 LSB | @ Port 5 and TUE_{max} = | ±10 LSB | @ PORT1 is guaranteed only if the absolute sum of input overload currents on all analog input pins does not exceed 10 mA.

Due to an internal problem, the specified TUE value is only met for a **positive** overload current 0 mA ≤ I_{OV} ≤ +5 mA (all currents flowing into the microcontroller are defined as positive and all currents flowing out of it are defined as negative).

If the exceptional conditions in the application system cause a **negative** overload current, then the maximum TUE can be exceeded (depending on value of I_{OV}, R_{AIN}, R_{AREF} and R_{AGND}):

Port 5 Problem Description in Detail:

1. Overload Current at analog channel AN0 - AN15

If a **negative** overload current I_{OV} occurs on analog input channel ANn, then an additional current I_{AN} (crosstalk current) is caused at the neighbour channels ANn-1 and ANn+1. This behavior causes an additional unadjusted error AUE to the ADC result.

Relation between I_{AN} and I_{OV}:

$$I_{ANn+1} = ovf\text{-}1 * I_{OVn}$$
$$I_{ANn-1} = ovf\text{-}1 * I_{OVn}$$

2. Overload Current at digital channel P7.7

A negative overload current I_{OV} at digital Port P7.7 causes an additional current I_{AN} at the analog input AN0. The relation between both channels is also defined by ovf-1:

$$I_{AN0} = ovf\text{-}1 * I_{OVP7.7}$$

3. Overload Current at analog channel AN12 – AN15 and influence to V_{AREF}

If an overload current I_{OV} occurs on analog input channel AN12 – AN15 then an additional current I_{AREF} (crosstalk current) is caused at pin V_{AREF}.
Depending on R_{AREF}, the internal resistance of the reference voltage, the crosstalk current I_{AREF} at pin V_{AREF} can cause an additional unadjusted error AUE to all other analog channels.
In case $R_{AREF} \leq$ 2450 Ohm [$R_{AREF} \leq$ ((LSB/2) / (I_{OVmax} * ovf-3)] the maximum possible additional error to all other channels is smaller than 0.5 LSB with the condition of I_{OVmax} = | ±5 mA| at AN12 - AN15.

Relation between I_{AREF} and I_{OV}:

$$I_{AREF} = ovf\text{-}3 * I_{OVn} \qquad (12 \leq n \leq 15)$$

Note: The influence to the reference voltage V_{AREF} caused by I_{OVn} (shift of V_{AREF}) is maximum for V_{AINn} = V_{AREF} and the influence is minimum for V_{AINn} = 0V. The condition $R_{AREF} \leq$ 2450 Ohm and 0.5 LSB is calculated for the worst case at V_{AINn} = V_{AREF}.

4. Values of ovf-1 and ovf-3

Parameter	Symbol	Min	Max
Overload factor-1	ovf-1	- 0.0001	0
Overload factor-3	ovf-3	- 0.0002	0

These Values are the absolute **maximum values measured in the lab and not tested!**

5. Effects on the Conversion Result and TUE

The effect on the conversion result and the TUE has to be calculated based on the crosstalk current and the impedance of the analog source R_{ASRC}. I_{ANn} causes an external voltage $U_{\Delta n}$ at the analog channel ANn (same principle for V_{AREF} and V_{AGND}) which is the reason for an additional unadjusted error **AUE** of the conversion result. This AUE can increase the specified total unadjusted error TUE (specified value: TUE = ± 2 LSB). The voltage $U_{\Delta n}$ is nearly independent on the voltage value of the analog source.

$U\Delta n = I_{ANn} * R_{ASRC}$
$U_{\Delta REF} = I_{AREF} * R_{AREF}$
$U_{\Delta GND} = I_{AGND} * R_{AGND}$
AUE = U∆n / 1 LSB [U∆n in mV and LSB in mV]
TUE = (± 2 LSB) ± AUE

Note: A negative overload current I_{OVn} decreases the analog signal voltage V_{AINn} and causes a negative AUE.

6. Calculation Example

Assumed system values:

I_{OV4} = -1 mA negative overload current at P5.4
R_{ASRC} = 20 kOhm resistance of the external sensor at P5.5
V_{AREF} = 5 V 1 LSB = 4.9 mV
R_{AREF} has to be considered for I_{OV} at ANn ($12 \le n \le 15$)
R_{AREF} = 500 Ohm ➔ V_{AREF} shift error is negligible

$I_{AN5} = ovf1 * I_{OV4}$
$I_{AN5} = (-1.0*E-4) * (- 1 mA)$
$I_{AN5} = 0.10\ \mu A$

$U\Delta 5 = I_{AN5} * R_{ASRC}$
$U\Delta 5 = 0.10\ \mu A * 20\ kOhm$
$U\Delta 5 = 2.0\ mV$

AUE = U∆n / 1 LSB
AUE = 2.0 mV / 4.9 mV
AUE = 0.41 LSB

Result: The negative overload current I_{OV4} of this system example can distort the real result of AN5 by an additional unadjusted error, $0\ LSB \le AUE \le 0.41\ LSB$.
The TUE is in the range of $-2.41\ LSB \le TUE \le 2.0\ LSB$.

PORT1 Problem Description in Detail:

1. **Overload Current at analog channel AN16 – AN23**
If a **negative** overload current I_{OV} occurs on analog input channel ANn, then an additional current I_{AN} (crosstalk current) is caused at the neighbour channels ANn-1 and ANn+1. This behavior causes an additional unadjusted error AUE to the ADC result.

Relation between I_{AN} and I_{OV}:
$I_{ANn+1} = ovf-11 * I_{OVn}$
$I_{ANn-1} = ovf-11 * I_{OVn}$

2. **Overload Current at digital channel P0H.7**
A negative overload current I_{OV} at digital Port P0H.7 causes an additional current I_{AN} at the analog input AN16. The relation between both channels is also defined by ovf-11:

$I_{AN0} = ovf-11 * I_{OVP0H.7}$

3. Values of ovf-11

Parameter	Symbol	Min	Max
Overload factor-11	ovf-11	- 0.0008	0

This value is the absolute **maximum value measured in the lab and is not tested!**

History List (since C167CS-4RM/LM device step (E)ES-AA)

Functional Problems

Functional Problem	Short Description	Fixed in step
PWRDN.1	Execution of PWRDN Instruction while pin NMI# = high	
SYSCON1.1	Reset Value in Register SYSCON1	
(POWER.19	SW- or WDT-Reset while SDD mode is active	see RST.9)
POWER.16	SDD and Sleep Mode cannot be terminated	
POWER.15	Wake Up from Sleep Mode not possible in Prescaler Mode	
POWER.10	Sleep Mode: CAN and ADC Modules enabled	ES-AB
POWER.8	Termination of Sleep Mode with HW Reset	(E)ES-AA
BUS.18	PEC transfers after JMPR	(E)ES-AA
BUS.19	Unlatched Chip Selects at Entry into Hold Mode	
CPU.21	BFLDL/BFLDH Instructions after Write Operations to internal IRAM	
RST.13	Power-up with Missing Clock	ES-AB
RST.11	Missing clock at XTAL1	ES-AB
RST.9.2	Software/Watchdog/Short HW reset during Slow Down operation with PLL off	
RST.8	Clock failure detection during external reset	
RST.7	Latching of P0 Reset Configuration when pin EA# = high	(E)ES-AA
RST.5	HW Reset not detected in PLL Mode during Slow Down operation when PLL is switched off	(E)ES-AA
CAPCOM.4	SW Access to P1H Overwrites CAPCOM HW Settings	
ADC.11	Modifications of ADM field while bit ADST = 0	
TCOMP.2	Pad Driver Temperature Compensation does not work properly (AA-steps)	ES-AB
TCOMP.3	Pad Driver Temperature Compensation disabled (AB-steps)	
CAN.9	Contents of Message Objects and Mask of Last Message Registers after Reset	
CAN.7	Unexpected Remote Frame Transmission	
X9	Read Access to XPERs in Visible Mode	
X17	XBUS Access after External Access with EWEN = 1 and/or BSWC = 1	

AC/DC Deviations

AC/DC Deviation	Short Description	Fixed in step
AC.t34.4	t34 (min): CLKOUT rising to ALE falling edge relaxed to -2ns - included in specification for tc11: -3 .. +6 ns	see Data Sheet
DC.VILS.2	Input low voltage (special threshold) V_{ILS} (max) relaxed to 1.8 V	
DC.HYS.2	Hysteresis (special threshold) relaxed to 200 mV	
AC.PLL.1	PLL Jitter > 3.8 % for fcpu < 12 MHz	see PLL.3.4
(PLL.3.1	Increased PLL Jitter caused by External Access	see PLL.3.4)
PLL.3.4	Increased PLL Jitter caused by External Access (modified formula)	
DC.IIDX.1	Idle Mode Supply Current 65 mA @ 25 MHz	see DC.IID_PD.1
DC.IID_PD.1	Idle and Power Down Mode Supply Currents: not tested	
ADCC.2.4	ADC Overload Current (AA-steps only)	see ADCC.2.5
ADCC.2.5	ADC Overload Current (not for AA-steps)	

Documentation Update - Modifications of Features

(reference: C167CS User's Manual V2.0, 2000-07)

- **Registers P5DIDIS, P1DIDIS not implemented**

In the Ax-steps of the C167CS-4RM/LM, registers P5DIDIS and P1DIDIS are **not** implemented

- **Single Chip Mode Reset** (p. 20-20 to p. 20-24)

Note: the **Single Chip Mode Reset** Feature will be abandoned in future steps of the C167CS derivatives. These steps will always read the reset configuration from PORT0, independent of the state (low or high) of pin EA#. However, the possibility to modify the configuration via register RSTCON will still be provided. See also the Application Note AP1637 'Reset and System Startup Configuration via PORT0 or Register RSTCON' on
http://www.infineon.com/cmc_upload/migrated_files/document_files/Application_Notes/ap163703.pdf

This change ensures hardware compatibility with existing systems which are based on the C167CS-32F up to step CB, C167CS-LM, C167CR-xx (all steps), and C167CS-4RM step BA or higher.

Application Hints

Handling of the SSC Busy Flag (SSCBSY)

In master mode of the High-Speed Synchronous Serial Interface (SSC), when register SSCTB has been written, flag SSCBSY is set to '1' when the baud rate generator generates the next internal clock pulse. The maximum delay between the time SSCTB has been written and flag SSCBSY=1 is up to 1/2 bit time. SSCBSY is cleared 1/2 bit time after the last latching edge.

When polling flag SSCBSY after SSCTB has been written, SSCBSY may not yet be set to '1' when it is tested for the first time (in particular at lower baud rates). Therefore, e.g. the following alternative methods are recommended:

1. test flag SSCRIR (receive interrupt request) instead of SSCBSY (in case the receive interrupt request is not serviced by CPU interrupt or PEC), e.g.

```
loop:    BCLR SSCRIR             ;clear receive interrupt request flag
         MOV SSCTB, #xyz         ;send character
wait_tx_complete:
         JNB SSCRIR, wait_tx_complete ;test SSCRIR
         JB SSCBSY, wait_tx_complete  ;test SSCBSY to achieve original
                                      timing(SSCRIR may be set 1/2 bit
                                      time before SSCBSY is cleared)
```

2. use a software semaphore bit which is set when SSCTB is written and is cleared in the SSC receive interrupt routine

Note on Interrupt Register behaviour of the CAN module

Due to the internal state machine of the CAN module, a specific delay has to be considered between resetting INTPND and reading the updated value of INTID. See Application Note AP2924 " Interrupt Register behaviour of the CAN module in Siemens 16-bit Microcontrollers" on

http://www.infineon.com/cmc_upload/migrated_files/document_files/Application_Notes/ap292401.pdf

Bootstrap Loader: Baudrate Detection in Single Chip Boot Mode

In single chip boot mode (Pin EA# = 1) the bootstrap loader mode is detected during hardware reset when pin RD# is tied low.

In this mode fcpu = fosc /2 (default configuration) and can be controlled via register RSTCON. In case of a desired PLL mode e.g. with a PLL factor of 5, the internal frequency fcpu in bootstrap loader mode is 10 times slower as in normal running mode. So a communication with an external host starts with a baudrate related to fcpu = fosc /2. It is recommended to establish a 2nd level loader that adapts fcpu and baudrate to a convenient transmission rate.

Oscillator Watchdog and Prescaler Mode

The OWD replaces the missing oscillator clock signal with the PLL clock (base frequency).

- In **direct drive** mode the PLL base frequency is used directly (fcpu = 2...5 MHz).
- In **prescaler** mode the PLL base frequency is divided by 2 (fcpu = 1...2.5 MHz).

(see also C167CS User's Manual V2.0, p.6-8)

Maintenance of ISNC register

The RTC and PLL interrupts share one interrupt node (XP3IC). If an interrupt request occurs the request bit in the Interrupt Subnode Control register has to be checked and cleared by software. To avoid a conflict with the next hardware interrupt request it is recommended to first clear both the request and the enable bit, and then to set the enable bit again.

Example for a RTC interrupt service routine (for Tasking C compiler):

```
...
If (RTCIR)
      {
      _bfld (ISNC, 0x0003, 0x0000); // clear RTCIE and RTCIR
      _putbit (1, ISNC, 1);         // set RTCIE
      }
...
```

In assembly language:

```
...
  EXTR      #1
  JNB       RTCIR, no_rtc_request
  EXTR      #2          ; no further interruption of this sequence possible
  BFLDL     ISNC, #03h, #00h  ; clear RTCIE and RTCIR
  BSET      RTCIE             ; set RTCIE
...
no_rtc_request:
```

Type_RE Oscillator Configuration for Low Power Operation

The type_RE main oscillator is optimized for oscillation with a crystal within a frequency range of 4...40 MHz. In order to achieve minimum power consumption in power saving modes where the oscillator remains active, the size of the external components (Cx1, Cx2, Rx2) of a crystal oscillator circuit may be reduced below the values which are listed in AP2420xx 'Crystal Oscillator of the C500 and C166 Microcontroller Families' on
http://www.infineon.com/cmc_upload/migrated_files/document_files/Application_Notes/ap242005.pdf

The recommended minimum values for the external circuits are equal to the values which are listed in AP2420xx for the Type_LP2 oscillator.

It is strongly recommended to measure the resulting safety factor (negative resistance method) in the final target system (layout) to ensure sufficient start-up reliability. Please refer to the limits specified by the crystal supplier.

PLL lock after temporary clock failure

When the PLL is locked and the input clock at XTAL1 is interrupted then the PLL becomes unlocked, provides the base frequency (2 ... 5 MHz) and the PLL unlock interrupt request flag is set. If the XTAL1 input clock starts oscillation again then the PLL stays in the PLL base frequency. The CPU clock source is only switched back to the XTAL1 oscillator clock after a hardware reset. This can be achieved via a normal hardware reset or via a software reset with enabled bidirectional reset. It is important that the hardware reset is at least active for 1 ms, after that time the PLL is locked in any case.

Application Support Group, Munich
http://www.infineon.com/cgi/ecrm.dll/ecrm/scripts/prod_cat.jsp?oid=-8137