Errata Sheet

Rel. 1.2, 2019-03

Device	**XMC1400**
Marking/Step	**EES-AA, ES-AA, AA**
Package	**PG-VQFN-40/48/64, PG-LQFP-64, PG-TSSOP-38**

Overview

This “Errata Sheet” describes product deviations with respect to the user documentation listed below.

Table 1 Current User Documentation

Document	Version	Date
XMC1400 Reference Manual AA-step	V1.1	Aug. 2016
XMC1400 Data Sheet AA-step	V1.3	Sep. 2016

Make sure that you always use the latest documentation for this device listed in category “Documents” at **http://www.infineon.com/xmc1000**.

Notes

1. *The errata described in this sheet apply to all temperature and frequency versions and to all memory size and configuration variants of affected devices, unless explicitly noted otherwise.*
2. *Devices marked with EES or ES are engineering samples which may not be completely tested in all functional and electrical characteristics, therefore they must be used for evaluation only. The specific test conditions for EES and ES are documented in a separate “Status Sheet”.*

Conventions used in this Document

Each erratum is identified by **Module_Marker**.**TypeNumber**:

- **Module**: Subsystem, peripheral, or function affected by the erratum.
- **Marker**: Used only by Infineon internal.
- **Type**: type of deviation
  - **(none)**: Functional Deviation
  - **P**: Parametric Deviation
  - **H**: Application Hint
  - **D**: Documentation Update
- **Number**: Ascending sequential number. As this sequence is used over several derivatives, including already solved deviations, gaps inside this enumeration can occur.

1 History List / Change Summary

Table 2 History List

Version	Date	Remark
1.2	2019-03	This Document. Changes see column “Chg” in the table below.
1.1	2016-11	
1.0	2015-10	Initial version

Table 3 Errata fixed in this step

Errata	Short Description	Change
- none -		

Table 4 Functional Deviations

Functional Deviation	Short Description	XMC1401	XMC1402	XMC1403	XMC1404	Chg	Pg
ACMP_CM.001	Operating range of the Analog Comparator Reference Divider function		X		X		8
ADC_AI.008	Wait-for-Read condition for register GLOBRES not detected in continuous auto-scan sequence	X	X	X	X		8
ADC_AI.016	No Channel Interrupt in Fast Compare Mode with GLOBRES	X	X	X	X		9
BCCU_CM.008	Linear walk starts with a delay after an aborted linear walk		X		X		9

Table 4 Functional Deviations

Functional Deviation	Short Description	XMC1401	XMC1402	XMC1403	XMC1404	Chg	Pg
BCCU_CM.009	Dimming level not immediately changed for first dimming operation		X		X		10
CCU_AI.007	Automatic shadow transfer feature does not work when system PCLK is faster than MCLK	X	X	X	X		10
CCU_AI.008	Clock ratio limitation when using MCSS inputs	X	X	X	X		11
CCU8_AI.005	PWM outputs from Compare Channel 2 are disabled in asymmetric mode		X		X		12
CCU8_AI.006	Timer concatenation does not work when using external count signal		X		X		13
CPU_CM.002	Watchpoint PC functions can report false execution	X	X	X	X		15
CPU_CM.003	Prefetch faulting instructions can erroneously trigger breakpoints	X	X	X	X		17
Firmware_CM.002	Calculate Target Level for Temperature Comparison User Routine returns zero for valid temperature input parameter	X	X	X	X		17

Table 4 Functional Deviations

Functional Deviation	Short Description	XMC1401	XMC1402	XMC1403	XMC1404	Chg	Pg
Firmware_CM.003	BMI installation of CAN BSL with external oscillator is not supported			X	X		18
Firmware_CM.004	SSC BSL is not supported	X	X	X	X		18
Firmware_CM.005	Last byte of SRAM is not available for ASC BSL	X	X	X	X		18
Firmware_CM.006	Header resend not supported for incorrect ASC BSL header byte	X	X	X	X		19
Firmware_CM.007	User ROM function _NvmProgVerifyBlock is non-functional for parts of the flash address space	X	X	X	X	New	19
SCU_CM.022	Transition to internal DCO1 clock after system reset fails if device is clocked by OSC_HP	X	X	X	X		20
SCU_CM.023	Reset of XTAL oscillator watchdog using XOWDRES does not work as specified	X	X	X	X	New	21
USIC_AI.008	SSC delay compensation feature cannot be used	X	X	X	X		21
USIC_AI.014	No serial transfer possible while running capture mode timer	X	X	X	X		22
USIC_AI.017	Clock phase of data shift in SSC slave cannot be changed	X	X	X	X		22

Table 4 Functional Deviations

Functional Deviation	Short Description	XMC1401	XMC1402	XMC1403	XMC1404	Chg	Pg
USIC_AI.018	Clearing PSR.MSLS bit immediately deasserts the SELOx output signal	X	X	X	X		22
WDT_CM.001	No overflow is generated for WUB default value	X	X	X	X		23

Table 5 Application Hints

Hint	Short Description	XMC1401	XMC1402	XMC1403	XMC1404	Chg	Pg
ACMP_CM.H002	Disabling the ORC	X	X	X	X		24
ADC_AI.H007	Ratio of Sample Time t_S to SHS Clock f_{SH}	X	X	X	X		24
BCCU_CM.H001	Additional dimming clocks after dimming curve switch		X		X		25
BCCU_CM.H004	Packer threshold (CHCONFIGy.PKTH) accepted values		X		X		25
BCCU_CM.H005	Enable a dimming engine for global dimming		X		X		26
MultiCAN_AI.H005	TxD Pulse upon short disable request			X	X		26
MultiCAN_AI.H006	Time stamp influenced by resynchronization			X	X		26

Table 5 Application Hints

Hint	Short Description	XMC1401	XMC1402	XMC1403	XMC1404	Chg	Pg
MultiCAN_AI.H007	Alert Interrupt Behavior in case of Bus-Off			X	X		26
MultiCAN_TC.H003	Message may be discarded before transmission in STT mode			X	X		27
MultiCAN_TC.H004	Double remote request			X	X		28
SCU_CM.H001	Temperature Sensor Functionality	X	X	X	X		28
USIC_AI.H004	I2C slave transmitter recovery from deadlock situation	X	X	X	X		29

Table 6 Documentation Updates

Hint	Short Description	XMC1401	XMC1402	XMC1403	XMC1404	Chg	Pg
ADC_CM.D001	Definition of trigger bus bits in register OCS is wrong	X	X	X	X		30
SCU_CM.D002	XTAL oscillator watchdog status output validity	X	X	X	X	New	30
WDT_CM.D001	Correction to section "Pre-warning Mode"	X	X	X	X		31

2 Functional Deviations

The errata in this section describe deviations from the documented functional behavior.

ACMP_CM.001 Operating range of the Analog Comparator Reference Divider function

The Analog Comparator Reference Divider function is not available when V_{DDP} is below 3 V. To use this function, V_{DDP} must be between 3 V to 5.5 V.

Workaround

None

ADC_AI.008 Wait-for-Read condition for register GLOBRES not detected in continuous auto-scan sequence

In the following scenario:

- A continuous auto-scan is performed over several ADC groups and channels by the Background Scan Source, using the global result register (GLOBRES) as result target (GxCHCTRy.RESTBS=1_B), and
- The Wait-for-Read mode for GLOBRES is enabled (GLOBRCR.WFR=1_B),

each conversion of the auto-scan sequence has to wait for its start until the result of the previous conversion has been read out of GLOBRES.

When the last channel of the auto-scan is converted and its result written to GLOBRES, the auto-scan re-starts with the highest channel number of the highest ADC group number. But the start of this channel does not wait until the result of the lowest channel of the previous sequence has been read from register GLOBRES, i.e. the result of the lowest channel may be lost.

Workaround

If either the last or the first channel in the auto-scan sequence does not write its result into GLOBRES, but instead into its group result register (selected via bit

GxCHCTRy.RESTBS=0_B), then the Wait-for-Read feature for GLOBRES works correctly for all other channels of the auto-scan sequence.

For this purpose, the auto-scan sequence may be extended by a “dummy” conversion of group x/ channel y, where the Wait-for-Read mode must not be selected (GxRCRy.WFR=0_B) if the result of this “dummy” conversion is not read.

ADC_AI.016 No Channel Interrupt in Fast Compare Mode with GLOBRES

In fast compare mode, the compare value is taken from bitfield RESULT of the global result register GLOBRES and the result of the comparison is stored in the respective bit FCR.

The channel event indicating that the input becomes higher or lower than the compare value, is not generated.

The comparison is executed correctly, the target bit is stored correctly, source events and result events are generated.

Workaround

The result bit FCR can be evaluated if the input is higher or lower than the compare value.

BCCU_CM.008 Linear walk starts with a delay after an aborted linear walk

If a linear walk is previously aborted, the subsequent linear walk starts with a delay. The maximum delay is one linear clock.

Workaround

None.

BCCU_CM.009 Dimming level not immediately changed for first dimming operation

For the first dimming operation, the dimming level is not immediately incremented or decremented upon a shadow bit (DES) assertion.

Workaround

None.

CCU_AI.007 Automatic shadow transfer feature does not work when system PCLK is faster than MCLK

Each CCU8 or CCU4 Timer Slice has a set of registers that have an associated shadow transfer mechanism. This shadow transfer mechanism works like a buffer where the application software can load new data, and when needed, request the load of this data into the register (this mechanism of loading new data into the current used register is called shadow transfer).

The shadow transfer can be applied to several data registers: compare values (that control the PWM duty cycle), timer period value or dead-time, in case of CCU8 (CCU4 does not have a dead-time feature).

To request the shadow transfer operation, the application software has two possibilities:

- do a software request by writing a 1_B into a specific bitfield
- or enable an automatic request feature, that will allow the hardware to request the shadow transfer in an automatic way (saving this way one software operation)

The automatic shadow transfer request be enabled for the following conditions:

- after the software writes a value into the shadow period register - enabled by setting the CC8ySTC.ASPC = 1_B (in CCU8) or CC4ySTC.ASPC = 1_B (in CCU4)
- after the software writes a value into the shadow compare register - enabled by setting the CC8ySTC.ASCC1 = 1_B (in CCU8) or CC8ySTC.ASCC2 = 1_B (in CCU8) or CC4ySTC.ASCC = 1_B (in CCU4)

- after the software writes a value into the passive level register - enabled by setting the CC8ySTC.ASLC = 1_B (in CCU8) or CC4ySTC.ASLC = 1_B
- after the software writes a value into the dither shadow register - enabled by setting the CC8ySTC.ASDC = 1_B (in CCU8) or CC4ySTC.ASDC = 1_B
- after the software writes a value into the floating prescaler shadow register - enabled by setting the CC8ySTC.ASFC = 1_B (in CCU8) or CC4ySTC.ASFC = 1_B

None of these automatic shadow transfer features work when the system PCLK frequency is higher than the MCLK frequency.

The PCLK can, in XMC1400 device(s), be configured to be 2x faster than the MCLK by setting the CLKCR.PCLKSEL = 1_B.

Whenever the PCLKSEL = 1_B, then none of the automatic shadow transfer request functions work.

Note that the rest of the shadow transfer mechanism/features are still usable as specified.

Workaround

None

CCU_AI.008 Clock ratio limitation when using MCSS inputs

The MCSS input signals of CCU8 and CCU4 units are erroneously sampled with the AHB bus clock f_{PERIPH} instead of the module clock f_{CCU}.

Implication

If the f_{PERIPH} and f_{CCU} frequencies are programmed to a ratio different from 1:1 then the MCSS signals running from POSIF to CCU4/CCU8 are not correctly sampled by CCU8/CCU4.

This can for example affect brushless DC motor drive applications when a clock ratio different from 1:1 is required.

Workaround

None

CCU8_AI.005 PWM outputs from Compare Channel 2 are disabled in asymmetric mode

Each CCU8 Timer Slice offers an asymmetric PWM mode. This mode is enabled whenever the CC8yCHC.ASE bitfield is set to 1_B.

If the asymmetric mode is not selected (ASE = 0_B), then each compare channel works independently, and can generate each, two complementary PWM signals. The PWM signal information for compare channel 1 is stored in the CC8yST1. The PWM signal information for compare channel 2 is stored in CC8yST2. CC8yST1 and CC8yST2 can then be forward to the PWM outputs. This is depicted in **Figure 1** a).

If the asymmetric mode is selected, the timer slice will generate two complementary asymmetric PWM signals, using the compare channel 1 for setting the PWM and the compare channel 2 for clearing the PWM. The asymmetric PWM signal is stored in CC8yST1 and then can be forward to the outputs. This is depicted in **Figure 1** b).

The asymmetric mode has a bug, where the PWM signal of Compare Channel 2 is not stored in CC8yST2. This means that from the Compare Channel 2 the timer cannot generate any PWM signal - depicted by the cross on **Figure 1** b).

Note that the asymmetric PWM is still stored in CC8yST1 and can be used normally by the application.

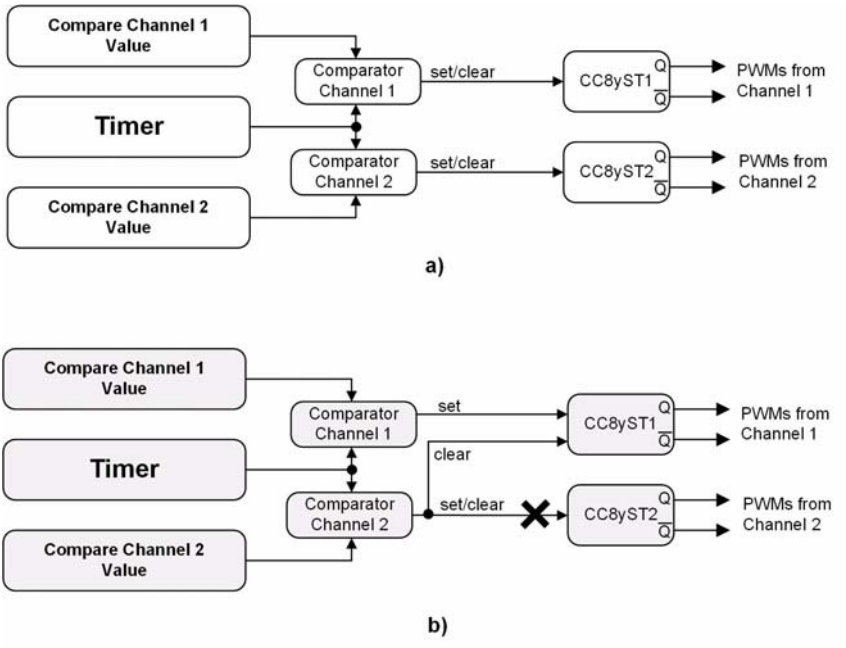

Figure 1 CCU8 simplifier PWM generation scheme - a) normal mode; b) asymmetric mode

Workaround

None

CCU8_AI.006 Timer concatenation does not work when using external count signal

Each CCU8 peripheral contains four sixteen bit timers. It is possible nevertheless to concatenate multiple timers to achieve a timer/counter with 32, 48 or 64 bits. To enable the concatenation feature, the CC8yCMC.TCE bitfield needs to be set to 1_B - Figure 1 a), where CCU8x represents a CCU8 peripheral instance x, and CC80 and CC81, represents timer 0 and timer 1 respectively (please notice that CC80 and CC81 are just used for simplicity, meaning that this function can be used also with the other timers inside CCU8x).

It is also possible to use an external signal as a count trigger. This means that when using an external count signal, the LSB timer is incremented each time that a transition on this external signal occurs - Figure 1 b).

When an external count signal is enabled - by programming the CC8yCMC.CNTS with 01_B, 10_B or 11_B - the concatenation function does not work. One cannot use in parallel the timer concatenation and external count signal features.

Note: On Figure 1, the count signal is used in CCU80 because this timer represents the LSBs. While the count signal could be enabled in the MSB timer (CC81), this does not make sense when the timers are concatenate - because the count should be used to increment the LSB timer. The LSB timer will then in each wrap around, increment the MSB timer.

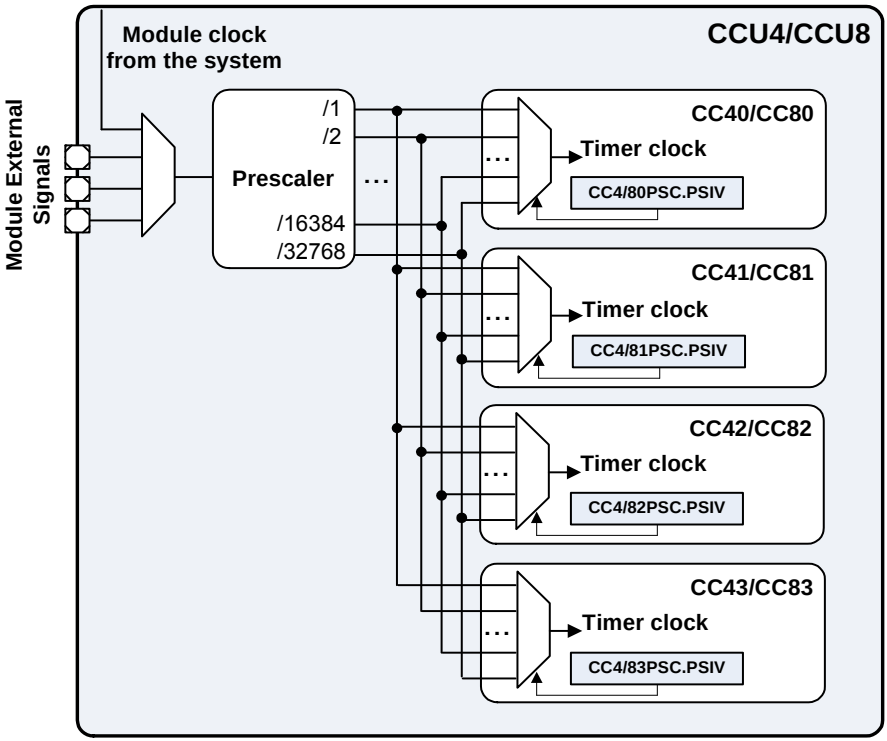

Figure 2 CCU8x concatenation feature resource configuration - a) without external count function; b) with external count function

Workaround

None

CPU_CM.002 Watchpoint PC functions can report false execution

In the presence of interrupts including those generated by the SVC instruction, it is possible for both the data watchpoint unit's PC match facility and PC

sample-register to operate as though the instruction immediately following the interrupted or SVC instruction had been executed.

Conditions

Either:

1. Halting debug is enabled via C_DEBUGEN = 1
2. Watchpoints are enabled via DWTENA = 1
3. A watchpoint is configured for PC sampling DWT_FUNCTION = 0x4
4. The same watchpoint is configured to match a `target instruction`
5. And either:
   a) The `target instruction` is interrupted before execution, or
   b) The `target instruction` is preceded by a taken SVC instruction
6. The DWT will unexpectedly match the `target instruction`
7. The processor will unexpectedly enter debug state once inside the exception handler

Or:

1. The debugger performs a read access to the DWT_PCSR
2. A `non-committed instruction` is preceded by a taken SVC instruction
3. The DWT_PCSR value unexpectedly matches the `non-committed instruction`

Implications

If halting debug is enabled and PC match watchpoints are being used, then spurious entry into halted debug state may occur under the listed conditions.

If the DWT_PCSR is being used for coarse grain profiling, then it is possible that the results can include hits for the address of an instruction immediately after an SVC instruction, even if said instruction is never executed.

Workaround

This errata does not impact normal execution of the processor.

A debug agent may choose to handle the infrequent false positive Debug state entry and erroneous PCSR values as spurious events.

CPU_CM.003 Prefetch faulting instructions can erroneously trigger breakpoints

External prefetch aborts on instruction fetches on which a BPU breakpoint has been configured, will cause entry to Debug state. This is prohibited by revision C of the ARMv6-M Architecture Reference Manual. Under this condition, the breakpoint should be ignored, and the processor should instead service the prefetch-abort by entering the HardFault handler.

Conditions

1. Halting debug is enabled via CDEBUG_EN == '1'
2. A BPU breakpoint is configured on an instruction in the first 0.5GB of memory
3. The fetch for said instruction aborts via an AHB Error response
4. The processor will erroneously enter Debug state rather than entering HardFault.

Implications

If halting debug is enabled and a BPU breakpoint is placed on an instruction with faults due to an external abort, then a non-compliant entry to Debug state will occur.

Workaround

This errata does not impact normal execution of the processor.

A debug agent may choose to avoid placing BPU breakpoints on addresses that generate AHB Error responses, or may simply handle the Debug state entry as a spurious debug event.

Firmware_CM.002 Calculate Target Level for Temperature Comparison User Routine returns zero for valid temperature input parameter

In Calculate Target Level for Temperature Comparison User Routine in Firmware, the temperature sensor threshold value is expected to be returned for a valid range of temperature input parameter of 233K to 388K. This user

function typically returns zero value for input parameter out of the valid range, also for some input parameters within the valid range.

Workaround

If user function returns zero for input parameter within the valid range, increase or decrease the input parameter by 1 degree Kelvin in order to use this user function.

Firmware_CM.003 BMI installation of CAN BSL with external oscillator is not supported

For the start-up mode selection of CAN BSL and CAN BSL with timeout mode, the Request BMI Installation User Routine with CAN clock source via external oscillator is not supported for EES samples.

Workaround

Use CAN clock source via internal oscillator in the Request BMI Installation User Routine for the CAN BSL start-up modes.

Firmware_CM.004 SSC BSL is not supported

SSC Bootstrap Loader start-up mode is not supported for EES and ES samples.

Workaround

None.

Firmware_CM.005 Last byte of SRAM is not available for ASC BSL

For start-up mode selection of ASC BSL and ASC BSL with timeout modes, the last byte of available SRAM is not available for user application code.
Application length error (BSL_NOK) is transmitted back to the host for application code length of available SRAM size.

Workaround

Host to send application code length of available SRAM size - 1 Byte or less.

Firmware_CM.006 Header resend not supported for incorrect ASC BSL header byte

For start-up mode selection of ASC BSL and ASC BSL with timeout modes, the baud rate detection is performed based on the Start Byte (00_H) received from the host. Next, the Header Byte defined in **Table 7** is expected from the host. An incorrect Header Byte received leads to a hang-up of the device.

Table 7 Header Byte definition in ASC BSL

Name	Length, Byte	Value	Description
Data sent by the Host:			
BSL_ASC_F	1	$6C_H$	Header requesting full duplex ASC mode with the current baud rate
BSL_ASC_H	1	12_H	Header requesting half duplex ASC mode with the current baud rate
BSL_ENC_F	1	93_H	Header requesting full duplex ASC mode with a request to switch the baud rate
BSL_ENC_H	1	ED_H	Header requesting half duplex ASC mode with a request to switch the baud rate

Workaround

None.

Firmware_CM.007 User ROM function _NvmProgVerifyBlock is non-functional for parts of the flash address space

Several user NVM related routines are available inside the XMC1400 ROM. The user routine “Program & Verify Flash Block” (_NvmProgVerifyBlock) in the

Boot ROM returns NVM_E_DST_ALIGNMENT, when bit 4 of parameter dstAddr is set to 1.

Therefore, only half of the address space can be programmed and verified with this ROM function.

Workaround

Option 1: Instead of using _NvmProgVerifyBlock, the XMC1400 _NvmProgVerify ROM function can be used to program the flash in pages (16 (blocks) * 4 (4 words per block) * 4 (4 bytes per word) = 256bytes).

Option 2: The user can implement their own XMC1400_NvmProgVerifyBlock. Detailed information about programming the NVM of XMC1400 can be found inside the user manual chapter “10.9. Example Sequences”.

SCU_CM.022 Transition to internal DCO1 clock after system reset fails if device is clocked by OSC_HP

If the device is clocked externally and configured to run from High Precision Oscillator Circuit (OSC_HP) and a system reset is triggered by hardware or software then the device will remain unclocked.

The problem occurs because with each system reset a transition to internal DCO1 clock is forced which does not correctly function in the described case.

Implications

If the device is running from High Precision Oscillator Circuit (OSC_HP) then the below workarounds shall be applied to handle system resets.

Workarounds

For software resets the following options are available:

- Switch clock to DCO1 before applying the system reset
- Trigger a master reset by setting bit RSTCON.MRSTEN to 1

For hardware (and software) resets the following option is available:

- Enable the internal watchdog timer (WDT). The WDT is running from the separate clock source DCO2 and therefore not affected by the problem.

After expiry of the timer WDT will trigger another reset which resolves the blocking situation.

SCU_CM.023 Reset of XTAL oscillator watchdog using XOWDRES does not work as specified

The XTAL oscillator watchdog (XOWD) monitors the external OSC_HP clock frequency using the standby clock as the reference clock. It can be enabled or disabled via OSCCSR.XOWDEN. By setting bit OSCCSR.XOWDRES, the detection for the external OSC_HP clock frequency can be restarted.

Register bit OSCCSR.XOWDRES alone is not working to clear and restart oscillator watchdog.

Workaround

To clear and restart the XTAL oscillator watchdog, the following sequence has to be executed:

1. Disable oscillator watchdog by setting XOWDEN = 0_B.
2. Wait for at least one standby clock cycle (+10% safety margin).
   One standby clock cycle = 1/32.768 kHz = ~30.52 us

Enable oscillator watchdog by setting XOWDEN = 1_B and XOWDRES = 1_B in one write access.

3. Wait until XOWDRES is set back to 0_B by hardware.

USIC_AI.008 SSC delay compensation feature cannot be used

SSC master mode and complete closed loop delay compensation cannot be used. The bit DX1CR.DCEN should always be written with zero to disable the delay compensation.

Workaround

None.

USIC_AI.014 No serial transfer possible while running capture mode timer

When the capture mode timer of the baud rate generator is enabled (BRG.TMEN = 1) to perform timing measurements, no serial transmission or reception can take place.

Workaround

None.

USIC_AI.017 Clock phase of data shift in SSC slave cannot be changed

Setting PCR.SLPHSEL bit to 1 in SSC slave mode is intended to change the clock phase of the data shift such that reception of data bits is done on the leading SCLKIN clock edge and transmission on the other (trailing) edge.

However, in the current implementation, the feature is not working.

Workaround

None.

USIC_AI.018 Clearing PSR.MSLS bit immediately deasserts the SELOx output signal

In SSC master mode, the transmission of a data frame can be stopped explicitly by clearing bit PSR.MSLS, which is achieved by writing a 1 to the related bit position in register PSCR.

This write action immediately clears bit PSR.MSLS and will deassert the slave select output signal SELOx after finishing a currently running word transfer and respecting the slave select trailing delay (T_{td}) and next-frame delay (T_{nf}).

However in the current implementation, the running word transfer will also be immediately stopped and the SELOx deasserted following the slave select delays.

If the write to register PSCR occurs during the duration of the slave select leading delay (T_{ld}) before the start of a new word transmission, no data will be transmitted and the SELOx gets deasserted following T_{td} and T_{nf}.

Workaround

There are two possible workarounds:

- Use alternative end-of-frame control mechanisms, for example, end-of-frame indication with TSCR.EOF bit.
- Check that any running word transfer is completed (PSR.TSIF flag = 1) before clearing bit PSR.MSLS.

WDT_CM.001 No overflow is generated for WUB default value

The Window Watchdog Timer (WDT) does not generate an overflow event if the default counter value $FFFFFFFF_H$ is used in register WUB.

Implications

Without an timer overflow no reset or pre-warning is requested. For other WUB values the WDT operates correctly and a reset or pre-warning is requested upon WDT overflow.

Workaround

Do not use $FFFFFFFF_H$ as counter value.

3 Application Hints

The errata in this section describe application hints which must be regarded to ensure correct operation under specific application conditions.

ACMP_CM.H002 Disabling the ORC

When the ORC is disabled while detecting an overvoltage, i.e. while its output signal is high, the output will not always return to zero, but may remain high after disabling.

It is therefore recommended to disable the connected units (ERU, interrupt generation) before disabling the corresponding out-of-range comparator.

ADC_AI.H007 Ratio of Sample Time t_S to SHS Clock f_{SH}

The sample time t_S is programmable to the requirements of the application.

To ensure proper operation of the internal control logic, t_S must be at least four cycles of the prescaled converter clock f_{SH}, i.e. $t_S \geq 4\ t_{CONV}$ x (DIVS+1).

(1) With **SHS*_TIMCFGx.SST > 0**, the sample time is defined by

t_S = SST x t_{ADC}.

In this case, the following relation must be fulfilled:

- SST ≥ 4 x t_{CONV}/t_{ADC} x (DIVS+1), i.e. SST ≥ 4 x f_{ADC}/f_{CONV} x (DIVS+1).
  - Example:
    with the default setting DIVS=0 and $f_{ADC} = f_{MCLK}$ = 32 MHz, $f_{SH} = f_{CONV}$ = 32 MHz (for DIVS = 0):
    select SST ≥ 4.

(2) With **SHS*_TIMCFGx.SST = 0**, the sample time is defined by

t_S = (2+STC) x t_{ADCI}, with $t_{ADCI} = t_{ADC}$ x (DIVA+1)

In this case, the following relation must be fulfilled:

- [(2+STC) x (DIVA+1)] / (DIVS+1) ≥ 4 x t_{CONV}/t_{ADC} = 4 x f_{ADC}/f_{CONV}.
  - Example:
    With the default settings STC=0, DIVA=1, DIVS=0 and $f_{ADC} = f_{MCLK}$ =

32 MHz, $f_{SH} = f_{CONV}$ = 32 MHz (for DIVS = 0),
this relation is fulfilled.

Note: In addition, the condition $f_{ADC} = f_{MCLK} \geq 0.55\ f_{SH}$ must be fulfilled. Note that this requirement is more restrictive than the requirement in ADC_AI.H006.

Definitions

DIVA: Divider Factor for the Analog Internal Clock, resulting from bit field GLOBCFG.DIVA (range: $1..32_D$)

DIVS: Divider Factor for the SHS Clock, resulting from bit field SHS*_SHSCFG.DIVS (range: $1..16_D$)

STC: Additional clock cycles, resulting from bit field STCS/STCE in registers GxICLASS*, GLOBICLACSSy (range: $0..256_D$)

SST: Short Sample Time factor, resulting from bit field SHS*_TIMCFGx.SST (range: $1..63_D$)

Recommendation

Select the parameters such that the sample time t_S is at least four cycles of the prescaled converter clock f_{SH}, as described above.

BCCU_CM.H001 Additional dimming clocks after dimming curve switch

If the dimming curve is switched (from coarse to fine or vice versa), the next dimming process takes additional dimming clocks.

BCCU_CM.H004 Packer threshold (CHCONFIGy.PKTH) accepted values

CHCONFIGy.PKTH is defined as 3-bits wide. However, only values 1-4 are accepted.

BCCU_CM.H005 Enable a dimming engine for global dimming

When using global dimming as the source of dimming input (CHCONFIG.DSEL = 111_B), enable at least one of the dimming engines (DEEN != 0).

MultiCAN_AI.H005 TxD Pulse upon short disable request

If a CAN disable request is set and then canceled in a very short time (one bit time or less) then a dominant transmit pulse may be generated by MultiCAN module, even if the CAN bus is in the idle state.

Example for setup of the CAN disable request:
`CAN_CLC.DISR=1` and then `CAN_CLC.DISR=0`

Workaround

Set all INIT bits to 1 before requesting module disable.

MultiCAN_AI.H006 Time stamp influenced by resynchronization

The time stamp measurement feature is not based on an absolute time measurement, but on actual CAN bit times which are subject to the CAN resynchronization during CAN bus operation.The time stamp value merely indicates the number of elapsed actual bit times. Those actual bit times can be shorter or longer than nominal bit time length due to the CAN resynchronization events.

Workaround

None.

MultiCAN_AI.H007 Alert Interrupt Behavior in case of Bus-Off

The MultiCAN module shows the following behavior in case of a bus-off status:

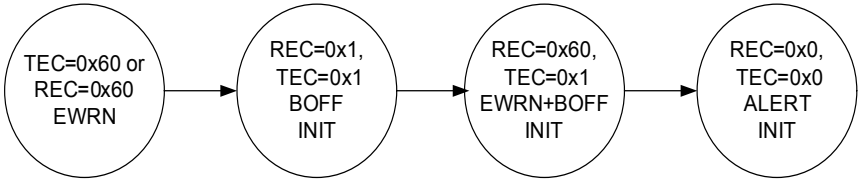

Figure 3 Alert Interrupt Behavior in case of Bus-Off

When the threshold for error warning (EWRN) is reached (default value of Error Warning Level EWRN = 0x60), then the EWRN interrupt is issued. The bus-off (BOFF) status is reached if TEC > 255 according to CAN specification, changing the MultiCAN module with REC and TEC to the same value 0x1, setting the INIT bit to 1_B, and issuing the BOFF interrupt. The bus-off recovery phase starts automatically. Every time an idle time is seen, REC is incremented. If REC = 0x60, a combined status EWRN+BOFF is reached. The corresponding interrupt can also be seen as a pre-warning interrupt, that the bus-off recovery phase will be finished soon. When the bus-off recovery phase has finished (128 times idle time have been seen on the bus), EWRN and BOFF are cleared, the ALERT interrupt bit is set and the INIT bit is still set.

MultiCAN_TC.H003 Message may be discarded before transmission in STT mode

If `MOFCRn.STT=1` (Single Transmit Trial enabled), bit TXRQ is cleared (TXRQ=0) as soon as the message object has been selected for transmission and, in case of error, no retransmission takes places.

Therefore, if the error occurs between the selection for transmission and the real start of frame transmission, the message is actually never sent.

Workaround

In case the transmission shall be guaranteed, it is not suitable to use the STT mode. In this case, `MOFCRn.STT` shall be 0.

MultiCAN_TC.H004 Double remote request

Assume the following scenario: A first remote frame (dedicated to a message object) has been received. It performs a transmit setup (TXRQ is set) with clearing NEWDAT. MultiCAN starts to send the receiver message object (data frame), but loses arbitration against a second remote request received by the same message object as the first one (NEWDAT will be set).

When the appropriate message object (data frame) triggered by the first remote frame wins the arbitration, it will be sent out and NEWDAT is not reset. This leads to an additional data frame, that will be sent by this message object (clearing NEWDAT).

There will, however, not be more data frames than there are corresponding remote requests.

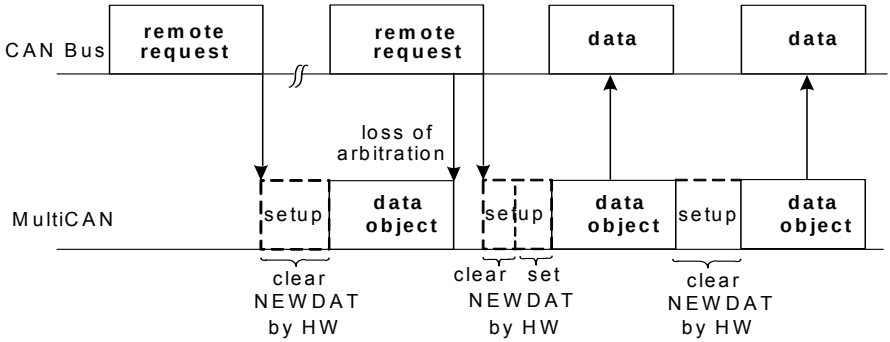

Figure 4 Loss of Arbitration

SCU_CM.H001 Temperature Sensor Functionality

EES samples are not temperature tested, therefore the temperature sensor functionality is not supported.

Workaround

None

USIC_AI.H004 I2C slave transmitter recovery from deadlock situation

While operating the USIC channel as an IIC slave transmitter, if the slave runs out of data to transmit before the master receiver issues clock pulses, for example due to an error in the application flow, it ties the SCL infinitely low.

Recommendation

To recover and reinitialize the USIC IIC slave from such a deadlock situation, the following software sequence can be used:

1. Switch the SCL and SDA port functions to be general port inputs for the slave to release the SCL and SDA lines:
   a) Write 0 to the two affected Pn_IOCRx.PCy bit fields.
2. Flush the FIFO buffer:
   a) Write 1_B to both USICx_CHy_TRBSCR.FLUSHTB and FLUSHRB bits.
3. Invalidate the internal transmit buffer TBUF:
   a) Write 10_B to USICx_CHy_FMR.MTDV.
4. Clear all status bits and reinitialize the IIC USIC channel if necessary.
5. Reprogram the Pn_IOCRx.PCy bit fields to select the SCL and SDA port functions again.

At the end of this sequence, the IIC slave is ready to communicate with the IIC master again.

4 Documentation Updates

The errata in this section contain updates to or completions of the user documentation. These updates are subject to be taken over into upcoming user documentation releases.

ADC_CM.D001 Definition of trigger bus bits in register OCS is wrong

The definition of register OCS contains bit fields controlling the OCDS trigger bus (OTBG). This trigger bus is not available in the product.

Correction

The following bit fields of register OCS are invalid:

- TGS
- TGB
- TG_P

The definition is changed to field “0”, type “r”, description “Reserved, write 0, read as 0”.

SCU_CM.D002 XTAL oscillator watchdog status output validity

Inside section “14.5.2.4 XTAL Oscillator Watchdog” of the reference manual it is specified, “…the detection status output will only be valid after **some** cycles of the standby clock frequency.”

Correction

These “some” cycles are precisely five cycles of the standby clock frequency.

Therefore it is recommended, to wait at least five standby clock cycles (+10% safety margin) before checking XTAL watchdog status for validity. Five standby clock cycles = 5 / 32.768kHz = ~152.6 us.

The sentence inside section 14.5.2.4 XTAL Oscillator Watchdog of the reference manual shall be re-phrased as follows:

“…the detection status output will only be valid after **five** cycles of the standby clock frequency.”

WDT_CM.D001 Correction to section "Pre-warning Mode"

Section “Pre-warning Mode” of WDT chapter in the Reference Manual states the following:

"… The alarm status is shown via register WDTSTS and can be cleared via register WDTCLR. A clear of the alarm status will bring the WDT back to normal state. The alarm signal is routed as request to the SCU, where it can be promoted to NMI. …"

Correction

The statement "A clear of the alarm status will bring the WDT back to normal state" is wrong.

A clear of the alarm status bit via write to WDTCLR.ALMC will clear only the bit WDSTSTS.ALMS.

To transfer the WDT back to the normal state a WDT service request is required.