MPC5646B_0N32E-0N40P

Mask Set Errata

Mask Set Errata for Mask 0N32E-0N40P

Revision History

This report applies to mask 0N32E-0N40P for these products:

- MPC5646B
- MPC5644B
- MPC5645B
- MPC5645C
- MPC5644C
- MPC5646C

Table 1. Revision History

Revision	Date	Significant Changes
30 SEP 2022	9/2022	The following errata were revised. • ERR011235 • ERR007394
28 FEB 2022	9/2022	The following errata were revised. • ERR050782 • ERR050575
30 JULY 2021	8/2021	The following errata were added. • ERR009682 • ERR003010 • ERR006350 • ERR050459 • ERR008970 • ERR050782 • ERR008770 • ERR011235 • ERR009764 • ERR009976 • ERR009978 • ERR010755 • ERR050575 • ERR011295 • ERR011294 • ERR011293

Table continues on the next page...

Table 1. Revision History (continued)

Revision	Date	Significant Changes
		• ERR008933 • ERR008951 • ERR007297 The following errata were revised. • ERR007394 • ERR004146
30 Nov 2015	11/2015	Initial Revision

Errata and Information Summary

Table 2. Errata and Information Summary

Erratum ID	Erratum Title
ERR007938	ADC: Possibility of missing CTU conversions
ERR004168	ADC: Abort switch aborts the ongoing injected channel as well as the upcoming normal channel
ERR003010	ADC: conversion chain failing after ABORT chain
ERR004186	ADC: Do not trigger ABORT or ABORTCHAIN prior to the start of CTU triggered ADC conversions and do not trigger ABORTCHAIN prior to the start of INJECTED triggered ADC conversions.
ERR005569	ADC: The channel sequence order will be corrupted when a new normal conversion chain is started prior to completion of a pending normal conversion chain
ERR006685	CFlash: Page buffer and prefetch not available for address range 0x0020_0000 to 0x002F_FFFF (upper 1MB)
ERR008227	CGM & ME: The peripheral set clock must be active during a peripheral clock enable or disable request
ERR008081	Debug – The e200z0 core instruction pointer may be corrupted when attaching the e200z0 to a debugger when the e200z0 system clock divider is configured in 2:1 mode.
ERR003556	DMA_MUX : Low Power Entry may not be completed when peripherals run on divided clock with DMA enabled mode
ERR009976	DSPI: Incorrect data received by master with Modified transfer format enabled when using Continuous serial communication clock mode
ERR006026	DSPI: Incorrect SPI Frame Generated in Combined Serial Interface Configuration
ERR010755	DSPI: Transmit and Receive FIFO fill flags in status register is not cleared when DMA is improperly configured
ERR050782	e200: Time Base TBU register contains wrong value during TBL rollover
ERR003697	e200z: Exceptions generated on speculative prefetch
ERR003512	ECSM: ECSM_PFEDR displays incorrect endianness

Table continues on the next page...

Table 2. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR006967	eDMA: Possible misbehavior of a preempted channel when using continuous link mode
ERR011235	EMIOS: Any Unified Channel running in OPWMB or OPWMCB mode may function improperly if the source counter bus is generated by Unified channel in MC mode
ERR050575	eMIOS: Any Unified Channel running in OPWMCB mode may function improperly if the lead or trail dead time insertion features is used and its timebase is generated by Unified channel in MCB mode
ERR011293	EMIOS: For any UC operating in OPWFMB mode the Channel Count register should not be written with a value greater than Channel B Data register value
ERR011295	EMIOS: In OPWFMB mode, A1/B1 registers do not get reloaded with A2/B2 register values if counter value returns 0x1 after counter wrap condition
ERR011294	EMIOS: OPWFMB and MCB mode counter rollover resets the counter to 0x0 instead of 0x1 as mentioned in the specification
ERR009978	eMIOS: Unexpected channel flag assertion during GPIO to MCB mode transition
ERR006620	FLASH: ECC error reporting is disabled for Address Pipelining Control (APC) field greater than Read Wait-State Control (RWSC) field.
ERR007322	FlexCAN: Bus Off Interrupt bit is erroneously asserted when soft reset is performed while FlexCAN is in Bus Off state
ERR008770	FlexRAY: Missing TX frames on Channel B when in dual channel mode and Channel A is disabled
ERR007732	FXOSC is active during standby mode if FIRC_CTL[FIRCDIV] bit is configured to be greater than 1
ERR008951	I2C: Attempting a start cycle while the bus is busy may generate a short clock pulse
ERR006082	LINFlexD : LINS bits in LIN Status Register(LINSR) are not usable in UART mode.
ERR004340	LINFlexD: Buffer overrun can not be detected in UART Rx FIFO mode
ERR008933	LINFlexD: Inconsistent sync field may cause an incorrect baud rate and the Sync Field Error Flag may not be set
ERR007297	LINFlexD: Response timeout values is loaded in LINOCR[OC2] field instead of LINOCR[OC1]
ERR008970	LINFlexD: Spurious bit error in extended frame mode may cause an incorrect Idle State
ERR007589	LINFlexD: Spurious timeout error when switching from UART to LIN mode or when resetting LINTCSR[MODE] bit in LIN mode
ERR006350	LINFlexD: WLS feature cannot be used in buffered mode
ERR007394	MC_ME: Incorrect mode may be entered on low-power mode exit.
ERR003570	MC_ME: Possibility of Machine Check on Low-Power Mode Exit
ERR003476	MC_RGM: Clearing a flag at RGM_DES or RGM_FES register may be prevented by a reset
ERR007953	ME: All peripherals that will be disabled in the target mode must have their interrupt flags cleared prior to target mode entry
ERR006481	NZ4C3/NZ7C3: Erroneous Resource Full Message under certain conditions

Table continues on the next page...

Table 2. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR007120	NZxC3: DQTAG implemented as variable length field in DQM message
ERR007688	RTC: An API interrupt may be triggered prematurely after programming the API timeout value
ERR009764	SARADC : DMA interface limitation depending on PBRIDGE/SARADC clock ratio
ERR004146	SARADC: Interrupted conversions are aborted, but may not be properly restored
ERR009682	SPI: Inconsistent loading of shift register data into the receive FIFO following an overflow event
ERR050459	SXOSC: Clock output may contain extra clock pulses in Normal mode

Known Errata

ERR007589: LINFlexD: Spurious timeout error when switching from UART to LIN mode or when resetting LINTCSR[MODE] bit in LIN mode

Description

If the LINFlexD module is enabled in Universal Asynchronous Receiver/Transmitter (UART) mode and the value of the MODE bit of the LIN Timeout Control Status register (LINTCSR) is 0 (default value after reset), any activity on the transmit or receive pins will cause an unwanted change in the value of the 8-bit field Output Compare Value 2 (OC2) of the LIN Output Compare register (LINOCR).

If the LINFlexD module is enabled in LIN mode and the value of the MODE bit of the LIN Timeout Control Status register (LINTCSR) is changed from '1' to '0', then the old value of the Output Compare Value 1 (OC1) and Output Compare Value 2 (OC2) of the LIN Output Compare register (LINOCR) is retained.

As a consequence, if the module is reconfigured from UART to Local Interconnect Network (LIN) mode, or LINTCSR MODE bit is changed from '1' to '0', an incorrect timeout exception is generated when the LIN communication starts.

Workaround

If the LINFlexD module needs to be switched from UART mode to LIN mode, before writing UARTCR[UART] to 1, ensure that the LINTCSR[MODE] is first set to 1.

If the LINFlexD module is in LIN mode and LINTCSR[MODE] needs to be switched from 1 to 0 in between frames, the LINOCR must be set to 0xFFFF by software.

ERR006620: FLASH: ECC error reporting is disabled for Address Pipelining Control (APC) field greater than Read Wait-State Control (RWSC) field.

Description

The reference manual states the following at the Platform flash memory controller Access pipelining functional description.

"The platform flash memory controller does not support access pipelining since this capability is not supported by the flash memory array. As a result, the APC (Address Pipelining Control) field should typically be the same value as the RWSC (Read Wait-State Control) field for best performance, that is, BKn_APC = BKn_RWSC. It cannot be less than the RWSC."

The reference manual advises that the user must not configure APC to be less than RWSC and typically APC should equal RWSC. However the documentation does not prohibit the configuration of APC greater than RWSC and for this configuration ECC error reporting will be disabled. Flash ECC error reporting will only be enabled for APC = RWSC.

For the case when flash ECC is disabled and data is read from a corrupt location the data will be transferred via the system bus however a bus error will not be asserted and neither a core exception nor an ECSM interrupt will be triggered. For the case of a single-bit ECC error the data will be corrected but for a double-bit error the data will be corrupt.

Notes

1. Both CFlash & DFlash are affected by this issue.

2. For single-bit and double-bit Flash errors neither a core exception nor an ECSM interrupt will be triggered unless APC=RWSC.

3. The Flash Array Integrity Check feature is not affected by this issue and will successfully detect an ECC error for all configurations of APC >= RWSC.

4. For the APC > RWSC configuration other than flash ECC error reporting there will be no other unpredictable behaviour from the flash.

5. The write wait-state control setting at PFCRx[BKn_WWSC] has no affect on the flash. It is recommend to set WWSC = RWSC = APC.

Workaround

PFCRx[BKy_APC] must equal PFCRx PFCRx[BKy_RWSC]. See datasheet for correct setting of RWSC.

ERR009682: SPI: Inconsistent loading of shift register data into the receive FIFO following an overflow event

Description

In the Serial Peripheral Interface (SPI) module, when both the receive FIFO and shift register are full (Receive FIFO Overflow Flag bit in Status Register is set (SR [RFOF] = 0b1)) and then the Clear Rx FIFO bit in Module Configuration Register (MCR [CLR_RXF]) is asserted to clear the receive FIFO, shift register data is loaded into the receive FIFO after the clear operation completes.

Workaround

1. Avoid a receive FIFO overflow condition (SR[RFOF] should never be 0b1). To do this, monitor the RX FIFO Counter field of the Status Register (SR[RXCTR]) which indicates the number of entries in receive FIFO and clear before the counter equals the FIFO depth.

2. Alternatively, after every receive FIFO clear operation (MCR[CLR_RXF] = 0b1) following a receive FIFO overflow (SR[RFOF] = 0b1) scenario, perform a single read from receive FIFO and discard the read data.

ERR003010: ADC: conversion chain failing after ABORT chain

Description

During a chain conversion while the ADC is in scan mode when ADC_MCR[ABORTCHAIN] is asserted the current chain will be aborted as expected. However, in the next scan the first conversion of the chain is performed twice and the last conversion of the chain is not performed.

Workaround

When aborting a chain conversion enable ADC_MCR[ABORTCHAIN] and disable ADC_MCR[START].

ADC_MCR[START] can be enabled when the abort is complete.

ERR006481: NZ4C3/NZ7C3: Erroneous Resource Full Message under certain conditions

Description

The e200zx core Nexus interface may transmit an erroneous Resource Full Message (RFM) following a Program Trace history message with a full history buffer. The History information for both of the messages are the same and the RFM should have not been transmitted. This occurs when the instruction following the indirect branch instruction (which triggers the Program Trace History message) would have incremented the History field. The instructions must be executed in back to back cycles for this problem to occur. This is the only case that cases this condition.

Workaround

There are three possible workarounds for this issue.

(1) Tools can check to see if the Program Trace History message and proceeding Resource Full Message (RFM) have the same history information. If the history is the same, then the RFM is extraneous and can be ignored.

(2) Code can be rewritten to avoid the History Resource Full Messages at certain parts of the code. Insert 2 NOP instructions between problematic code. Or inserting an "isync" or a indirect branch some where in the code sequence to breakup/change the flow.

(3) If possible, use Traditional Program Trace mode can be used to avoid the issue completely. However, depending on other conditions (Nexus port width, Nexus Port speed, and other enabled trace types), overflows of the port could occur.

ERR003570: MC_ME: Possibility of Machine Check on Low-Power Mode Exit

Description

When executing from the flash and entering a Low-Power Mode (LPM) where the flash is in low-power or power-down mode, 2-4 clock cycles exist at the beginning of the RUNx to LPM transition during which a wakeup or interrupt will generate a checkstop due to the flash not being available on RUNx mode re-entry. This will cause either a checkstop reset or machine check interrupt.

Workaround

If the application must avoid the reset, two workarounds are suggested:

1) Configure the application to handle the machine check interrupt in RAM dealing with the problem only if it occurs

2) Configure the MCU to avoid the machine check interrupt, executing the transition into low power modes in RAM

There is no absolute requirement to work around the possibility of a checkstop reset if the application can accept the reset, and associated delays, and continue. In this event, the WKPU.WISR will not indicate the channel that triggered the wakeup though the F_CHKSTOP flag will indicate that the reset has occurred. The F_CHKSTOP flag could still be caused by other error conditions so the startup strategy from this condition should be considered alongside any pre-existing strategy for recovering from an F_CHKSTOP condition.

ERR004168: ADC: Abort switch aborts the ongoing injected channel as well as the upcoming normal channel

Description

If an Injected chain (jch1,jch2,jch3) is injected over a Normal chain (nch1,nch2,nch3,nch4) the Abort switch does not behave as expected.

Expected behavior:

Correct Case (without SW Abort on jch3): Nch1- Nch2(aborted) -Jch1 - Jch2 - Jch3 - Nch2(restored) - Nch3 - Nch4

Correct Case(with SW Abort on jch3): Nch1 - Nch2(aborted) -Jch1 - Jch2 - Jch3(aborted) - Nch2(restored) - Nch3 - Nch4

Observed unexpected behavior:

Fault1 (without SW abort on jch3): Nch1 - Nch2(aborted) - Jch1 - Jch2 - Jch3 - Nch3 - Nch4 (Nch2 not restored)

Fault2 (with SW abort on jch3): Nch1- Nch2 (aborted) - Jch1 - Jch2 - Jch3(aborted) - Nch4 (Nch2 not restored & Nch3 conversion skipped)

Workaround

It is possible to detect the unexpected behavior by using the CEOCFRx register. The CEOCFRx fields will not be set for a not restored or skipped channel, which indicates this issue has occurred. The CEOCFRx fields need to be checked before the next Normal chain execution (in scan mode). The CEOCFRx fields should be read by every ECH interrupt at the end of every chain execution.

ERR006350: LINFlexD: WLS feature cannot be used in buffered mode

Description

The Flexible Local Interconnect Network (LINFlex) module may not operate correctly if the Special Word Length (WLS) for enabling 12-bit data length is selected in the Universal Asynchronous Receiver/Transmitter (UART) Mode Control Register (UARTCR) and the module is configured in the receive buffered mode.

Workaround

When WLS mode is required, always use the First In, First Out (FIFO) mode of the UART LINFLEX module by setting the Receive FIFO/Buffer mode bit of the UARTCR (UARTCR[RFBM]=1). In addition, the UART word length bits must be set (UARTCR[WL0 = 0b1] and UARTCR[WL1] = 0b1).

ERR050459: SXOSC: Clock output may contain extra clock pulses in Normal mode

Description

The 32 kHz Slow External Crystal Oscillator (SXOSC) may generate an extra clock pulse per clock period when configured in Normal mode using an external crystal. The SXOSC correctly generates positive clock edges aligned to the external crystal clock transitions but an extra clock pulse may be generated near the positive edge of the clock waveform. The SXOSC interface to the external crystal is not affected. This issue may only occur in the default Normal mode and operates as expected in Bypass mode (i.e. OSC_CTL[OSCBYP]=1).

The SXOSC clock source may be selected as the clock source for the Real Time Counter and Autonomous Periodic Interrupt (RTC/API) and also as the clock to be measured in the CMU Frequency Meter (via the CMU_FDR register). The RTC/API and CMU Frequency Meter counters may get an extra clock per SXOSC clock period causing a higher than expected count (affecting the calculated time or wakeup duration) or may theoretically cause a corrupted count value. The occurrence of the potential clock pulse may vary from clock period to clock period ranging from no extra clock pulse to one extra clock pulse per period. SXOSC may also be selected to be reflected to the CLKOUT pin. Worse case conditions to produce an extra clock pulse are lower temperatures.

Workaround

Use the SXOSC in Bypass mode instead of Normal mode by setting OSC_CTL[OSCBYP]. Bypass mode does not support an external crystal and thus an external clock source is needed.

Select other clock sources for the RTC/API and CMU Frequency Meter. The RTC/API supports other clock sources which include the 128 kHZ Slow Internal RC Oscillator (SIRC), 16 MHz Fast Internal RC Oscillator (FIRC), and 4-16 MHz Fast External Crystal Oscillator (FXOSC).

ERR006026: DSPI: Incorrect SPI Frame Generated in Combined Serial Interface Configuration

Description

In the Combined Serial Interface (CSI) configuration of the Deserial Serial Peripheral Interface (DSPI) where data frames are periodically being sent (Deserial Serial Interface, DSI), a Serial Peripheral Interface (SPI) frame may be transmitted with incorrect framing.

The incorrect frame may occur in this configuration if the user application writes SPI data to the DSPI Push TX FIFO Register (DSPI_PUSHR) during the last two peripheral clock cycles of the Delay-after-Transfer (DT) phase. In this case, the SPI frame is corrupted.

Workaround

Workaround 1: Perform SPI FIFO writes after halting the DSPI.

To prevent writing to the FIFO during the last two clock cycles of DT, perform the following steps every time a SPI frame is required to be transmitted:

Step 1: Halt the DSPI by setting the HALT control bit in the Module Configuration Register (DSPI_MCR[HALT]).

Step 2: Poll the Status Register's Transmit and Receive Status bit (DSPI_SR[TXRXS]) to ensure the DSPI has entered the HALT state and completed any in-progress transmission. Alternatively, if continuous polling is undesirable in the application, wait for a fixed time interval such as 35 baud clocks to ensure completion of any in-progress transmission and then check once for DSPI_SR[TXRXS].

Step 3: Perform the write to DSPI_PUSHR for the SPI frame.

Step 4: Clear bit DSPI_MCR[HALT] to bring the DSPI out of the HALT state and return to normal operation.

Workaround 2: Do not use the CSI configuration. Use the DSPI in either DSI-only mode or SPI-only mode.

Workaround 3: Use the DSPI's Transfer Complete Flag (TCF) interrupt to reduce worst-case wait time of Workaround 1.

Step 1: When a SPI frame is required to be sent, halt the DSPI as in Step 1 of Workaround 1 above.

Step 2: Enable the TCF interrupt by setting the DSPI DMA/Interrupt Request Select and Enable Register's Transmission Complete Request Enable bit (DSPI_RSER[TCF_RE])

Step 3: In the TCF interrupt service routine, clear the interrupt status (DSPI_SR[TCF]) and the interrupt request enable (DSPI_RSER[TCF_RE]). Confirm that DSPI is halted by checking DSPI_SR[TXRXS] and then write data to DSPI_PUSHR for the SPI frame. Finally, clear bit DSPI_MCR[HALT] to bring the DSPI out of the HALT state and return to normal operation.

ERR008970: LINFlexD: Spurious bit error in extended frame mode may cause an incorrect Idle State

Description

The LINFlexD module may set a spurious Bit Error Flag (BEF) in the LIN Error Status Register (LINESR), when the LINFlexD module is configured as follows:

- Data Size greater than eight data bytes (extended frames) by configuring the Data Field Length (DFL) bitfield in the Buffer Identifier Register (BIDR) with a value greater than seven (eight data bytes)

- Bit error is able to reset the LIN state machine by setting Idle on Bit Error (IOBE) bit in the LIN Control Register 2 (LINCR2)

As consequence, the state machine may go to the Idle State when the LINFlexD module tries the transmission of the next eight bytes, after the first ones have been successfully transmitted and Data Buffer Empty Flag (DBEF) was set in the LIN Status Register (LINSR).

Workaround

Do not use the extended frame mode by configuring Data Field Length (DFL) bit-field with a value less than eight in the Buffer Identifier Register (BIDR) (BIDR[DFL] < 8)

ERR050782: e200: Time Base TBU register contains wrong value during TBL rollover

Description

The e200 Time Base (TB) facility is a 64-bit structure provided for maintaining the time of day and operating interval timers. The TB consists of two 32-bit registers - time base upper (TBU) and time base lower (TBL). TBU and TBL are concatenated to provide a long-period 64-bit counter. TBL increments until its value becomes 0xFFFF_FFFF. The intended behavior is that at the next increment when the TBL value becomes 0x0000_0000 that the TBU value is incremented. But the actual behavior is that after the TBL value becomes 0x0000_0000, the TBU value will not increment until the transition of the TBL value to 0x0000_0001.

Workaround

Software will need to take care about the wrong TBU value during TBL rollover. Use the following sequence for reading TBU and TBL values:

loop:

mfspr r12, TBL

mfspr r3, TBU

mfspr r4, TBL

cmpl r4,r12

se_blt loop

ERR008770: FlexRAY: Missing TX frames on Channel B when in dual channel mode and Channel A is disabled

Description

If the FlexRay module is configured in Dual Channel mode, by clearing the Single Channel Device Mode bit (SCM) of the Module Control register (FR_MCR[SCM]=0), and Channel A is disabled, by clearing the Channel A Enable bit (FR_MCR[CHA]=0) and Channel B is enabled, by setting the Channel B enable bit (FR_MCR[CHB]=1), there will be a missing transmit (TX) frame in adjacent minislots (even/odd combinations in Dynamic Segment) on Channel B for certain communication cycles. Which channel handles the Dynamic Segment or Static Segment TX message buffers (MBs) is controlled by the Channel Assignment bits (CHA, CHB) of the Message Buffer Cycle Counter Filter Register (FR_MBCCFRn). The internal Static Segment boundary indicator actually only uses the Channel A slot counter to identify the Static Segment boundary even if the module configures the Static Segment to Channel B (FR_MBCCFRn[CHA]=0 and FR_MBCCFRn[CHB]=1). This results in the Buffer Control Unit waiting for a corresponding data acknowledge signal for minislot:N in the Dynamic Segment and misses the required TX frame transmission within the immediate next minislot:N+1.

Workaround

1. Configure the FlexRay module in Single Channel mode (FR_MCR[SCM]=1) and enable Channel B (FR_MCR[CHB]=1) and disable Channel A (FR_MCR[CHA]=0). In this mode the internal Channel A behaves as FlexRay Channel B. Note that in this mode only the internal channel A and the FlexRay Port A is used. So externally you must connect to FlexRay Port A.

2. Enable both Channel A and Channel B when in Dual Channel mode (FR_MCR[CHA=1] and FR_MCR[CHB]=1). This will allow all configured TX frames to be transmitted correctly on Channel B.

ERR003697: e200z: Exceptions generated on speculative prefetch

Description

The e200z4 core can prefetch up to 2 cache lines (64 bytes total) beyond the current instruction execution point. If a bus error occurs when reading any of these prefetch locations, the machine check exception will be taken. For example, executing code within the last 64 bytes of a memory region such as internal SRAM or FLASH, may cause a bus error when the core prefetches past the end of memory. An ECC exception can occur if the core prefetches locations that are valid, but not yet initialized for ECC.

Workaround

Do not place code to be executed within the last 64 bytes of a memory region. When executing code from internal ECC SRAM, initialize memory beyond the end of the code until the next 32-byte aligned address and then an additional 64 bytes to ensure that prefetches cannot land in uninitialized SRAM.

ERR011235: EMIOS: Any Unified Channel running in OPWMB or OPWMCB mode may function improperly if the source counter bus is generated by Unified channel in MC mode

Description

The Unified channel (UC) configured in Center Aligned Output Pulse Width Modulation Buffered (OPWMCB) or Output Pulse Width Modulation Buffered (OPWMB) modes is not working properly when it is sourced from the UC configured in Modulus Counter (MC) mode by setting the channel control register MODE bitfield to 0x10 or 0x11 and any of its pre-scalers (internal or global) divider ratio is higher than 1.

Workaround

When a counter bus is generated by the UC set in the MC mode with any pre-scaler (internal or global) divider ration higher than 1, don't use this counter bus for the UC set in OPWMCB or OPWMB mode.

ERR007394: MC_ME: Incorrect mode may be entered on low-power mode exit.

Description

For the case when the Mode Entry (MC_ME) module is transitioning from a run mode (RUN0/1/2/3) to a low power mode (HALT/STOP/STANDBY*) if a wake-up or interrupt is detected one clock cycle after the second write to the Mode Control (ME_MCTL) register, the MC_ME will exit to the mode previous to the run mode that initiated the low power mode transition.

Example correct operation DRUN->RUN1-> RUN3->STOP->RUN3

Example failing operation DRUN->RUN1-> RUN3->STOP->RUN1

Note STANDBY mode is not available on all MPC56xx microcontrollers

Workaround

To ensure the application software returns to the run mode (RUN0/1/2/3) prior to the low power mode (HALT/STOP/STANDBY*) it is required that the RUNx mode prior to the low power mode is entered twice.

The following example code shows RUN3 mode entry prior to a low power mode transition.

ME.MCTL.R = 0x70005AF0; /* Enter RUN3 Mode & Key */

ME.MCTL.R = 0x7000A50F; /* Enter RUN3 Mode & Inverted Key */

while (ME.GS.B.S_MTRANS) {} /* Wait for RUN3 mode transition to complete */

ME.MCTL.R = 0x70005AF0; /* Enter RUN3 Mode & Key */

ME.MCTL.R = 0x7000A50F; /* Enter RUN3 Mode & Inverted Key */

while (ME.GS.B.S_MTRANS) {} /* Wait for RUN3 mode transition to complete */

/* Now that run mode has been entered twice can enter low power mode */

/* (HALT/STOP/STANDBY*) when desired. */

ERR009764: SARADC : DMA interface limitation depending on PBRIDGE/SARADC clock ratio

Description

The Successive Approximation Register Analog-to-Digital Converter (SARADC) modules can trigger a Direct Memory Access (DMA) request through the DMA Enable (DMAE) register interface.

When the SARADC clock (SAR_CLK) frequency is slower than half of the peripheral bridge (PBRIDGEx_CLK) clock frequency, the SARADC may trigger a spurious transfer request to the DMA module after the completion of a first valid transfer.

Workaround

Setting the DMA clear sequence enable (DCLR) bit in the DMAE register (DMAE[DCLR] = 1) forces the clearing of the DMA request on read access to the data register and therefore prevents the spurious DMA transfer request.

In case the Internal Channel Data Registers (ICDRn) are only accessed through DMA module (i.e. there are no bus accesses to ICDRn registers triggered by other than DMA bus master when the DMAE[DMAEN] bit is set), it is possible to configure DMAE[DCLR] bit to '1'. This will clear DMA transfer request on the first DMA read access, ensuring both that DMA triggered transfer will complete successfully and that no other spurious DMA request will be triggered.

This work-around can be applied when any of below condition can be met:

- frequency ratio PBRIDGEx_CLK/SAR_CLK <= 8/3

- PBRIDGEx_CLK is 40MHz and SAR_CLK >= 14MHz

ERR009976: DSPI: Incorrect data received by master with Modified transfer format enabled when using Continuous serial communication clock mode

Description

When the Deserial Serial Peripheral Interface (DSPI) module is configured as follows:

1. Master mode is enabled (Master/Slave Mode Select bit in Module Configuration Register is set (DSPI_MCR [MSTR] = 0b1))

2. Modified transfer format is enabled (Modified Transfer Format Enable bit in Module Configuration Register is set (DSPI_MCR [MTFE] = 0b1))

3. Continuous serial communication clock mode is enabled (Continuous SCK Enable bit in Module Configuration Register is set (DSPI_MCR [CONT_SCKE] = 0b1))

In this configuration if the frame size of the current frame is greater than the frame size of the next received frame, corrupt frames are received in two scenarios:

a) Continuous Peripheral Chip Select Enable bit in PUSH TX FIFO Register is set (DSPI_PUSHR [CONT] = 0b1)

b) DSPI_PUSHR [CONT] = 0b0 and lower significant bit of the frame is transferred first (LSB first bit in Clock and Transfer Attributes Register is set (DSPI_CTAR [LSBFE] =0b1))

Workaround

To receive correct frames:

a) When DSPI_PUSHR [CONT] = 0b1, configure the frame size of the current frame less than or equal to the frame size of the next frame (for all frames).

b) When DSPI_PUSHR [CONT] = 0b0, configure DSPI_CTAR [LSBFE] = 0b0. Alternatively, configure the frame size of the current frame less than or equal to the frame size of the next frame (for all frames).

Make sure that for all received frames, the bits are read equal to their respective frame sizes and any extra bits during POP operation are masked.

ERR003556: DMA_MUX : Low Power Entry may not be completed when peripherals run on divided clock with DMA enabled mode

Description

System may not enter into Low Power Mode (HALT/STOP/STANDBY) when all the below conditions are true simultaneously:

1.A Peripheral with DMA capability is programmed to work on divided clock.

2. Above peripheral is programmed to be stopped in Low Power Mode and active in RUN Mode.

3. Above Peripheral is active with DMA transfer while Software requests change to Low Power mode.

Workaround

Software should ensure that all the DMA enabled peripherals have completed their transfer before requesting Low Power mode Entry

ERR009978: eMIOS: Unexpected channel flag assertion during GPIO to MCB mode transition

Description

When changing an Enhanced Modular IO Subsystem (eMIOS) channel mode from General Purpose Input/Output (GPIO) to Modulus Counter Buffered (MCB) mode, the channel flag in the eMIOS Channel Status register (eMIOS_Sn[FLAG]) may incorrectly be asserted. This will cause an unexpected interrupt or DMA request if enabled for that channel.

Workaround

In order to change the channel mode from GPIO to MCB without causing an unexpected interrupt or DMA request, perform the following steps:

(1) Clear the FLAG enable bit in the eMIOS Control register (eMIOS_Cn[FEN] = 0).

(2) Change the channel mode (eMIOS_Cn[MODE]) to the desired MCB mode.

(3) Clear the channel FLAG bit by writing '1' to the eMIOS Channel Status register FLAG field (eMIOS_Sn[FLAG] = 1).

(4) Set the FLAG enable bit (eMIOS_Cn[FEN] = 1) to re-enable the channel interrupt or DMA request reaction.

ERR008227: CGM & ME: The peripheral set clock must be active during a peripheral clock enable or disable request

Description

An individual peripheral clock can be enabled or disabled for a target mode via the Mode Entry Peripheral Control register (ME_PCTL) and the Mode Entry RUN/Low Power Peripheral Configuration register (ME_RUN_PC & ME_LP_PC). For this process to complete the user must ensure that the peripheral set clock relative to the specific peripheral is enabled for the duration of the current-mode-to-target-mode transition. The peripheral set clock is configured at the Clock Generation Module System Clock Divider Configuration Register (CGM_SC_DC).

A caveat for FlexCAN is for the case when the FXOSC is selected for the CAN Engine Clock Source (FLEXCAN_CTRL[CLK_SRC]). In this instance to enable or disable the FlexCAN peripheral clock the user must ensure FXOSC is enabled through the target mode transition i.e. FXOSC must be enabled for the target mode.

Workaround

To enable a peripheral clock:

1. Enable the peripheral set clock at CGM_SC_DC.

2. Enable the peripheral clock for the target mode at ME_PCTL & ME_RUN_PC/ ME_LP_PC.

3. Note steps 1 & 2 are interchangeable.

4. Transition to the target mode to enable the peripheral clock.

To disable a peripheral clock:

1. Disable the peripheral clock for the target mode at ME_PCTL & ME_RUN_PC/ ME_LP_PC.

2. Transition to the target mode to disable the peripheral clock.

3. Optionally disable peripheral set clock at CGM_SC_DC. Note to check other peripherals in this peripheral set are not required.

ERR006082: LINFlexD : LINS bits in LIN Status Register(LINSR) are not usable in UART mode.

Description

When the LINFlexD module is used in the Universal Asynchronous Receiver/Transmitter (UART) mode, the LIN state bits (LINS3:0]) in LIN Status Register (LINSR) always indicate the value zero. Therefore, these bits cannot be used to monitor the UART state.

Workaround

LINS bits should be used only in LIN mode.

ERR004146: SARADC: Interrupted conversions are aborted, but may not be properly restored

Description

When a triggered conversion interrupts an in process conversion in the Successive Approximation Analog to Digital Converter (SARADC), it is possible that the aborted conversion does not get restored to the SARADC and is not converted during the chain. Vulnerable configurations are:

- Injected chain over a normal chain
- Cross Trigging Unit (CTU) trigger over a normal chain
- CTU trigger over an injected chain

When any of these triggers arrive while the SARADC is in the conversion stage of the sample and conversion, the sample is discarded and is not restored. This means that the channel data register (SARADC_xCDRn) will not show the channel as being valid and the register SARADC_xCIPRn field CEOCFRx will not indicate a pending conversion. The sample that was aborted is lost.

If the injection occurs when the finite state machine switches from the sample phase, it is possible that on resuming normal chain, the chain is restored from an incorrect channel. This may lead to a second conversion on one of the channels in the chain.

When the trigger arrives during the final (last) channel conversion in a normal or injected chain, the same failure mode can cause two ECH (End of chain) interrupts to be raised in the interrupt register (SARADC_ISR).

If the trigger arrives during the sampling phase of the last channel in the chain, an ECH is triggered immediately, the trigger is processed and the channel is restored and after sampling/conversion, a second ECH interrupt occurs.

In scan mode, the second ECH does not occur if the trigger arrives during the conversion phase. In one-shot mode, a trigger arriving during the conversion phase of the last channel restarts the whole conversion chain and the next ECH occurs at completion of that chain.

Workaround

The application should check for valid data using the Channel Data Register, Internal Channel Data Register or Test Channel Data Register (CDR) status bits or the CEOCFRx registers to ensure all expected channels have converted. This can be tested by running a bitwise AND and an XOR with either the Channel Conversion Mask Register (JCMRx) or the Channel Conversion Mask Register (NCMRx and the CEOCFRx registers during the JECH handler. Any non-zero value for (xCMRx & (xCMRx $\oplus$ CEOCFRx)) / (SARADC_ICxCMRn & (SARADC_ICxCMRn $\oplus$ SARADC_IPICRn.EOC_CHx)) indicates that a channel has been missed and conversion should be requested again.

Spurious ECH interrupts can be detected by checking the NSTART/JSTART flags in the SARADC_MSR Module Status Registers – if the flag remains set during an ECH interrupt then another interrupt will follow after the restored channel or chain has been sampled and converted.

The spurious ECH workaround above applies to single-shot conversions. In single-shot mode, NSTART changes from 1 to 0. Therefore, the user can rely on checking the NSTART bit to confirm if a spurious ECH has occurred. However, for scan

mode, the NSTART bit will remain set during normal operation, so it cannot be relied upon to check for the spurious ECH issue. Consequently, if the CTU is being used in trigger mode, the conversions must be single-shot and not scan mode.

ERR006967: eDMA: Possible misbehavior of a preempted channel when using continuous link mode

Description

When using Direct Memory Access (DMA) continuous link mode Control Register Continuous Link Mode (DMA_CR[CLM]) = 1) with a high priority channel linking to itself, if the high priority channel preempts a lower priority channel on the cycle before its last read/write sequence, the counters for the preempted channel (the lower priority channel) are corrupted. When the preempted channel is restored, it continues to transfer data past its "done" point (that is the byte transfer counter wraps past zero and it transfers more data than indicated by the byte transfer count (NBYTES)) instead of performing a single read/write sequence and retiring.

The preempting channel (the higher priority channel) will execute as expected.

Workaround

Disable continuous link mode (DMA_CR[CLM]=0) if a high priority channel is using minor loop channel linking to itself and preemption is enabled. The second activation of the preempting channel will experience the normal startup latency (one read/write sequence + startup) instead of the shortened latency (startup only) provided by continuous link mode.

ERR010755: DSPI: Transmit and Receive FIFO fill flags in status register is not cleared when DMA is improperly configured

Description

The Deserial/Serial Peripheral Interface Transmit and Receive First In/First Out (FIFO) buffers can request additional information to be transferred via the Direct Memory Access (DMA) module when either the Transmit or Receive FIFO Fill/Drain Flags are set in the DSPI Status Register (SR[TFFF/RFDF]). However, the Transmit Fill Flag indicates that at least 1 location each (2 bytes each) in the Transmit and Command FIFOs is available to be written. It does not indicate that the FIFO is empty. Similarly, Receive FIFO fill flag only indicates at least 1 location (2 bytes) of the FIFO is available to be read. It does not indicate that the FIFO is full. If the DMA is configured to transfer more than 1 FIFO location size of data, the FIFO Fill/Drain Flags may not be properly cleared indicating that the FIFO is not full even when the FIFO is actually full (for Transmit FIFO) and not empty when the FIFO is actually empty (for Receive FIFO).

Workaround

Properly configure the DMA to fill/drain only 2 bytes to Transmit, Command and Receive FIFOs. Use the DMA loop to transfer more data if needed.

ERR008081: Debug – The e200z0 core instruction pointer may be corrupted when attaching the e200z0 to a debugger when the e200z0 system clock divider is configured in 2:1 mode.

Description

For the case when both e200z4 and e200z0 cores are active and the e200z4:e200z0 clocks are configured in 2:1 mode (by setting CGM_Z0_DCR[DIV] = 1), the e200z0 core instruction pointer may be corrupted when a debugger is attached to the e200z0 core.

Standalone operation of the device (no debugger attached) is not impacted.

Note – typically the NPC_PCR is written by the debug tool automatically when it connects to the MCU. The method of manually configuring the NPC_PCR will depend on the debugger vendor.

Workaround

Recommended configuration for JTAG (Nexus1 non-trace) z0 debug :

- NPC_PCR[MCKO_EN] = 0, NPC_PCR[EVTEN] = 0, and MCKO_DIV = /1, /4 or /8 (do not set /2)

Recommended configuration for Nexus3+ (trace) z0 debug:

- Configuration of NPC_PCR and CGM_Z0_DCR[DIV] (for 2:1 mode) should be done before z0 is released from reset by the z4

- NPC_PCR[MCKO_EN] = 1 and NPC_PCR [MCKO_DIV] = /4 or /8 (do not set /2)

- The NPC_PCR must be written by doing an "attach" with the z4 debugger

Note - NPC_PCR[MCKO_EN] = 1 & NPC_PCR[MCKO_DIV] = /2 is a valid configuration for Nexus3+ e200z4 debug when the e200z0 is not enabled or e200z4 and e200z0 are operating in 1:1 frequency mode.

ERR003476: MC_RGM: Clearing a flag at RGM_DES or RGM_FES register may be prevented by a reset

Description

Clearing a flag at RGM_DES and RGM_FES registers requires two clock cycles because of a synchronization mechanism. As a consequence if a reset occurs while clearing is on-going the reset may interrupt the clearing mechanism leaving the flag set.

Note that this failed clearing has no impact on further flag clearing requests.

Workaround

No workaround for all reset sources except SW reset.

Note that in case the application requests a SW reset immediately after clearing a flag in RGM_xES the same issue may occur. To avoid this effect the application must ensure that flag clearing has completed by reading the RGM_xES register before the SW reset is requested.

ERR005569: ADC: The channel sequence order will be corrupted when a new normal conversion chain is started prior to completion of a pending normal conversion chain

Description

If One shot mode is configured in the Main Configuration Register (MCR[MODE] = 0) the chained channels are automatically enabled in the Normal Conversion Mask Register 0 (NCMR0). If the programmer initiates a new chain normal conversion, by setting MCR[NSTART] = 0x1, before the previous chain conversion finishes, the new chained normal conversion will not follow the requested sequence of converted channels.

For example, if a chained normal conversion sequence includes three channels in following sequence: channel0, channel1 and channel2, the conversion sequence is started by MCR[NSTART] = 0x1. The software re-starts the next conversion sequence when MCR[NSTART] is set to 0x1 just before the current conversion sequence finishes.

The conversion sequence should be: channel0, channel1, channel2, channel0, channel1, channel2.

However, the conversion sequence observed will be: channel0, channel1, channel2, channel1, channel1, channel2. Channel0 is replaced by channel1 in the second chain conversion and channel1 is converted twice.

Workaround

Ensure a new conversion sequence is not started when a current conversion is ongoing. This can be ensured by issuing the new conversion setting MCR[NSTART] only when MSR[NSTART] = 0.

Note: MSR[NSTART] indicates the present status of conversion. MSR[NSTART] = 1 means that a conversion is ongoing and MSR[NSTART] = 0 means that the previous conversion is finished.

ERR050575: eMIOS: Any Unified Channel running in OPWMCB mode may function improperly if the lead or trail dead time insertion features is used and its timebase is generated by Unified channel in MCB mode

Description

The Unified channel (UC) configured in Center Aligned Output Pulse Width Modulation Buffered (OPWMCB) mode is not working properly when:

1. It's timebase is sourced from the UC configured in Modulus Counter Buffered (MCB) mode.

2. The lead or trail dead time insertion features is used.

3. Its channel prescaler is different than timebase channel prescaler.

Workaround

Channel configured in OPWMCB mode with lead or trail dead time insertion features enabled must have channel prescaler equal to the timebase channel prescaler configured in MCB mode.

ERR007938: ADC: Possibility of missing CTU conversions

Description

The CTU prioritizes and schedules trigger sources so that the ADC will receive only one CTU trigger at a time. However, whilst a Normal or Injected ADC conversion is ongoing as the ADC moves state from IDLE-to-SAMPLE , SAMPLE-to-WAIT, WAIT-to-SAMPLE and WAIT-to-IDLE there are 2 clock cycles at the state transition that a CTU trigger may be missed by the ADC.

Workaround

To ensure all CTU triggers are received at the ADC Normal and Injected modes must be disabled.

ERR011295: EMIOS: In OPWFMB mode, A1/B1 registers do not get reloaded with A2/B2 register values if counter value returns 0x1 after counter wrap condition

Description

In Output Pulse-Width and Frequency Modulation Buffered (OPWFMB) mode, A1/B1 registers do not get reloaded with A2/B2 register values if counter value returns 0x1 after counter wrap condition.

In order to avoid the counter wrap condition make sure internal counter value is within the 0x1 to B1 register value range when the OPWFMB mode is entered. Also overwriting of Channel Count register by forcing 'freeze' in OPWFMB mode should not take internal counter outside 0x1 to B register value.

Workaround

In order to avoid the counter wrap condition:

1. Make sure internal counter value is within the 0x1 to (B1 register) value range when the OPWFMB mode is entered.

2. Overwrite of Channel Count register by forcing 'freeze' in OPWFMB mode should not be outside the range of 0x1 to (B register) value.

ERR011294: EMIOS: OPWFMB and MCB mode counter rollover resets the counter to 0x0 instead of 0x1 as mentioned in the specification

Description

When the enhanced Modular Input/Output System (eMIOS) is used in Output Pulse-Width and Frequency Modulation Buffered (OPWFMB) or Modulus Counter Buffered (MCB) modes, when the counter rolls over, the counter returns to 0x0 instead of 0x1 as specified in the reference manual.

Workaround

In order to avoid the counter wrap condition:

1. Make sure internal counter value is within the 0x1 to (B1 register) value range when the OPWFMB mode is entered.

2. Overwrite of Channel Count register by forcing 'freeze' in OPWFMB mode should not be outside the range of 0x1 to (B register) value.

ERR006685: CFlash: Page buffer and prefetch not available for address range 0x0020_0000 to 0x002F_FFFF (upper 1MB)

Description

For the Cflash address range 0x0020_0000 to 0x002F_FFFF (upper 1MB) the Cflash page buffer and prefetch functions are not available. This affects all master accesses (e200z4d instruction port, e200z4d data port, e200z0h instruction port, e200z0h data port, eDMA, FEC, FlexRay and CSE) and affects both Cflash slave ports P0 (e200z4d instruction port) and P1 (all other masters). As no read to the affected Cflash region can be buffered all accesses will be delayed by the flash array access time as defined in the datasheet.

Workaround

It is recommended to place e200z4d cacheable code in the affected region 0x0020_0000 to 0x002F_FFFF (upper 1MB), as the cache shall negate the affect of the errata and performance impact should be minimal. The e200z0h core does not support cache and therefore code for the e200z0h core should be placed in the unaffected region 0x0000_0000 to 0x001F_FFFF (lower 2MB). It has been observed that by utilizing this workaround that there has been no degradation of e200z4d core performance. However, the user should be aware that it is difficult to quantify the affect that this errata will have on MCU performance, as it will depend on the flow of the software in the affected Cflash region.

ERR007688: RTC: An API interrupt may be triggered prematurely after programming the API timeout value

Description

When the API is enabled (RTCC[APIEN]), the API interrupt flag is enabled (RTCC[APIIE]) and the API timeout value (RTCC[APIVAL]) is programmed the next API interrupt may be triggered before the programmed API timeout value. Successive API Interrupts will be triggered at the correct time interval.

Workaround

The user must not use the first API interrupt for critical timing tasks.

ERR003512: ECSM: ECSM_PFEDR displays incorrect endianness

Description

The ECSM_PFEDR register reports ECC data using incorrect endiannes. For example, a flash location that contains the data 0xAABBCCDD would be reported as 0xDDCCBBAA at ECSM_PFEDR.

This 32-bit register contains the data associated with the faulting access of the last, properly-enabled flash ECC event. The register contains the data value taken directly from the data bus.

Workaround

Software must correct endianness.

ERR011293: EMIOS: For any UC operating in OPWFMB mode the Channel Count register should not be written with a value greater than Channel B Data register value

Description

For any Unified Channel (UC) running in Output Pulse-Width and Frequency Modulation Buffered (OPWFMB) mode, Channel Control Register MODE bitfield = 7'h1011000 or 7'h1011010, the internal counter runs from 0x1 to Channel B Data register value.

The internal counter can be overwritten by software using the Chanel Count register during 'freeze' operation.

If a counter wrap occurs due to overwriting of the counter with a value greater than its expiry value (B Data Register value); than the output signal behavior cannot be guaranteed.

Workaround

For any UC operating in OPWFMB mode the Channel Count register should not be written with a value greater than Channel B Data register value.

ERR007732: FXOSC is active during standby mode if FIRC_CTL[FIRCDIV] bit is configured to be greater than 1

Description

Irrespective of the ME_STANBY_MC[FXOSCON] bit configuration, FXOSC is active (powered ON) during standby mode if the FIRC_CTL[FIRCDIV] value is greater than 1.

Workaround

Do not configure FIRC_CTL[FIRDIV] value to be greater than one while switching from DRUN or RUN to Standby Mode.

ERR007120: NZxC3: DQTAG implemented as variable length field in DQM message

Description

The e200zx core implements the Data Tag (DQTAG) field of the Nexus Data Acquisition Message (DQM) as a variable length packet instead of an 8-bit fixed length packet. This may result in an extra clock ("beat") in the DQM trace message depending on the Nexus port width selected for the device.

Workaround

Tools should decode the DQTAG field as a variable length packet instead of a fixed length packet.

ERR004186: ADC: Do not trigger ABORT or ABORTCHAIN prior to the start of CTU triggered ADC conversions and do not trigger ABORTCHAIN prior to the start of INJECTED triggered ADC conversions.

Description

When ADC_MCR[ABORT] or ADC_MCR[ABORTCHAIN] is set prior to the ADC receiving a CTU trigger, the next CTU triggered ADC conversion will not be performed and further CTU triggered ADC conversions will be blocked.

When ADC_MCR[ABORTCHAIN] is set prior to the ADC receiving an INJECTED trigger, the next INJECTED ADC conversion will not be performed. Following the ABORTCHAIN command the MCU behaviour does not meet the specification as ADC_ISR[JECH] is not set and ADC_MCR[ABORTCHAIN] is not cleared.

Workaround

Do not program ADC_MCR[ABORT] or ADC_MCR[ABORTCHAIN] before the start of ADC conversions.

The case when CTU triggered ADC conversions are blocked should be avoided however it is possible to reactivate CTU conversions by clearing and setting ADC_MCR[CTUEN].

ERR007953: ME: All peripherals that will be disabled in the target mode must have their interrupt flags cleared prior to target mode entry

Description

Before entering the target mode, software must ensure that all interrupt flags are cleared for those peripheral that are programmed to be disabled in the target mode. A pending interrupt from these peripherals at target mode entry will block the mode transition or possibly lead to unspecified behaviour.

Workaround

For those peripherals that are to be disabled in the target mode the user has 2 options:

1. Mask those peripheral interrupts and clear the peripheral interrupt flags prior to the target mode request.

2. Through the target mode request ensure that all those peripheral interrupts can be serviced by the core.

ERR007322: FlexCAN: Bus Off Interrupt bit is erroneously asserted when soft reset is performed while FlexCAN is in Bus Off state

Description

Under normal operation, when FlexCAN enters in Bus Off state, a Bus Off Interrupt is issued to the CPU if the Bus Off Mask bit (CTRL[BOFF_MSK]) in the Control Register is set. In consequence, the CPU services the interrupt and clears the ESR[BOFF_INT] flag in the Error and Status Register to turn off the Bus Off Interrupt.

In continuation, if the CPU performs a soft reset after servicing the bus off interrupt request, by either requesting a global soft reset or by asserting the MCR[SOFT_RST] bit in the Module Configuration Register, once MCR[SOFT_RST] bit transitions from 1 to 0 to acknowledge the soft reset completion, the ESR[BOFF_INT] flag (and therefore the Bus Off Interrupt) is re-asserted.

The defect under consideration is the erroneous value of Bus Off flag after soft reset under the scenario described in the previous paragraph.

The Fault Confinement State (ESR[FLT_CONF] bit field in the Error and Status Register) changes from 0b11 to 0b00 by the soft reset, but gets back to 0b11 again for a short period, resuming after certain time to the expected Error Active state (0b00). However, this late correct state does not reflect the correct ESR[BOFF_INT] flag which stays in a wrong value and in consequence may trigger a new interrupt service.

Workaround

To prevent the occurrence of the erroneous Bus Off flag (and eventual Bus Off Interrupt) the following soft reset procedure must be used:

1. Clear CTRL[BOFF_MSK] bit in the Control Register (optional step in case the Bus Off Interrupt is enabled).

2. Set MCR[SOFT_RST] bit in the Module Configuration Register.

3. Poll MCR[SOFT_RST] bit in the Module Configuration Register until this bit is cleared.

4. Wait for 4 peripheral clocks.

5. Poll ESR[FLTCONF] bit in the Error and Status Register until this field is equal to 0b00.

6. Write "1" to clear the ESR[BOFF_INT] bit in the Error and Status Register.

7. Set CTRL[BOFF_MSK] bit in the Control Register (optional step in case the Bus Off Interrupt is enabled).

ERR008933: LINFlexD: Inconsistent sync field may cause an incorrect baud rate and the Sync Field Error Flag may not be set

Description

When the LINFlexD module is configured as follows:

1. LIN (Local Interconnect Network) slave mode is enabled by clearing the Master Mode Enable bit in the LIN Control Register 1 (LINCR1[MME] = 0b0)

2. Auto synchronization is enabled by setting LIN Auto Synchronization Enable (LINCR1[LASE] = 0b1)

The LINFlexD module may automatically synchronize to an incorrect baud rate without setting the Sync Field Error Flag in the LIN Error Status register (LINESR[SFEF]) in case Sync Field value is not equal to 0x55, as per the Local Interconnect Network (LIN) specification.

The auto synchronization is only required when the baud-rate in the slave node can not be programmed directly in software and the slave node must synchronize to the master node baud rate.

Workaround

There are 2 possible workarounds.

Workaround 1:

When the LIN time-out counter is configured in LIN Mode by clearing the MODE bit of the LIN Time-Out Control Status register (LINTCSR[MODE]= 0x0]):

1. Set the LIN state Interrupt enable bit in the LIN Interrupt Enable register (LINIER[LSIE] = 0b1)

2. When the Data Reception Completed Flag is asserted in the LIN Status Register (LINSR[DRF] = 0b1) read the LIN State field (LINSR[LINS])

3. If LINSR[LINS]= 0b0101, read the Counter Value field of the LIN Time-Out Control Status register (LINTCSR[CNT]), otherwise repeat step 2

4. If LINTCSR[CNT] is greater than 0xA, discard the frame.

When the LIN Time-out counter is configured in Output Compare Mode by setting the LINTCSR[MODE] bit:

1. Set the LIN State Interrupt Enable bit in the LIN Interrupt Enable register (LINIER[LSIE])

2. When the Data Reception Completed flag bit is asserted in the LIN Status Register (LINSR[DRF] = 0b1), read the LINSR[LINS] field

3. If LINSR[LINS]= 0b0101, store LINTCSR[CNT] value in a variable (ValueA), otherwise repeat step 2

4. Clear LINSR[DRF] flag by writing LINSR[LINS] field with 0xF

5. Wait for LINSR[DRF] to become asserted again and read LINSR[LINS] field

6. If LINSR[LINS] = 0b0101, store LINTCSR[CNT] value in a variable (ValueB), else repeat step 4

7. If ValueB - ValueA is greater than 0xA, discard the frame

Workaround 2:

Do not use the auto synchronization feature (disable with LINCR1[LASE] = 0b0) in LIN slave mode.

ERR008951: I2C: Attempting a start cycle while the bus is busy may generate a short clock pulse

Description

When the I2C (Inter-Integrated Circuit) is operating in a multi-master network and a start cycle is attempted by the I2C device when the bus is busy, the attempting master will lose arbitration as expected but a short extra clock cycle is generated in the bus. After losing arbitration, the master switches to slave mode but it does not detect the short clock pulse. The acknowledge signal is expected at the ninth clock by the current bus master but it is not sent as expected due to the undetected short clock pulse.

Workaround

Software must ensure that the I2C BUS is idle by checking the bus busy bit in the I2C Bus Status Register (I2C_IBSR.IBB) before switching to master mode and attempting a Start cycle.

ERR007297: LINFlexD: Response timeout values is loaded in LINOCR[OC2] field instead of LINOCR[OC1]

Description

In the LINFlex module, the response timeout value calculated by hardware is loaded onto the OC2[7:0] (Output Compare 2) bits of LINOCR (LIN Output Compare Register) instead of being loaded into the OC1[7:0] (Output Compare 1) bits of the same register as stated in the documentation.

This applies when the Time-out counter mode is enabled by clearing the MODE (Time-out counter mode) bit in the LINTCSR (LIN Timeout Control Status Register).

Workaround

Expect that OC2[7:0] (Output Compare 2) bits are loaded by hardware with the response timeout value when the MODE (Time-out counter mode) bit in LINTCSR is cleared.

ERR004340: LINFlexD: Buffer overrun can not be detected in UART Rx FIFO mode

Description

When the LINFlexD is configured in UART Receive (Rx) FIFO mode, the Buffer Overrun Flag (BOF) bit of the UART Mode Status Register (UARTSR) register is cleared in the subsequent clock cycle after being asserted.

User software can not poll the BOF to detect an overflow.

The LINFlexD Error Combined Interrupt can still be triggered by the buffer overrun. This interrupt is enabled by setting the Buffer Overrun Error Interrupt Enable (BOIE) bit in the LIN Interrupt enable register (LINIER). However, the BOF bit will be cleared when the interrupt routine is entered, preventing the user from identifying the source of error.

Workaround

Buffer overrun errors in UART FIFO mode can be detected by enabling only the Buffer Overrun Interrupt Enable (BOIE) in the LIN interrupt enable register (LINIER).

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Suitability for use in automotive applications — This NXP product has been qualified for use in automotive applications. If this product is used by customer in the development of, or for incorporation into, products or services (a) used in safety critical applications or (b) in which failure could lead to death, personal injury, or severe physical or environmental damage (such products and services hereinafter referred to as "Critical Applications"), then customer makes the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, safety, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. As such, customer assumes all risk related to use of any products in Critical Applications and NXP and its suppliers shall not be liable for any such use by customer. Accordingly, customer will indemnify and hold NXP harmless from any claims, liabilities, damages and associated costs and expenses (including attorneys' fees) that NXP may incur related to customer's incorporation of any product in a Critical Application.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.