Mask Set Errata for Mask 2M35Y

Introduction

This report applies to mask 2M35Y for these products:

- MPC563XM

Errata ID	Errata Title
3521	DECFIL: Soft reset failures at the end of filtering
6026	DSPI: Incorrect SPI Frame Generated in Combined Serial Interface Configuration
7352	DSPI: reserved bits in slave CTAR are writable
3378	EQADC: Pull devices on differential pins may be enabled for a short period of time during and just after POR
5642	ETPU2: Limitations of forced instructions executed via the debug interface
2740	ETPU2: Watchdog Status Register (WDSR) may fail to update on channel timeout
5640	ETPU2: Watchdog timeout may fail in busy length mode
3114	FLASH: Erroneous update of the ADR register in case of multiple ECC errors
3196	FLASH: PFCR3 is not directly writable
2379	FMPLL: Loss-of-clock detection may cause unexpected reset
5498	Flash: Prefetch during program/erase operation causes system bus stop
7322	FlexCAN: Bus Off Interrupt bit is erroneously asserted when soft reset is performed while FlexCAN is in Bus Off state
3407	FlexCAN: CAN Transmitter Stall in case of no Remote Frame in response to Tx packet with RTR=1
5909	MPC563xM/SPC563M: MIDR MASKNUM field is set to 0x22
7590	MPC563xM: Incorrect JTAG ID[MIC] and MIDR[S_F] register values
3205	NEXUS: EVTI not functional on QFP176 and BGA208 packages
6726	NPC: MCKO clock may be gated one clock period early when MCKO frequency is programmed as SYS_CLK/8.and gating is enabled
3425	PMC: 5V VDDREG POR De-assertion Max Level 4.2V
3221	PMC: SRAM standby power low voltage detect circuit is not accurate
2338	Pad Ring: Leakage if VDDE is greater than VDD33

Table continues on the next page...

Errata ID	Errata Title
3377	Pad Ring:Nexus pins may drive an unknown value immediately after power up but before the 1st clock edge
1421	SWT: switching SWT to system clock has very small chance of causing the SWT to enter an indeterminate state
4480	eQADC: Differential conversions with 4x gain may halt command processing
5086	eQADC: unexpected result may be pushed when Immediate Conversion Command is enabled
1297	eSCI : reads of the SCI Data Register, which clears the RDRF flag, may cause loss of frame if read occurs during reception of the STOP bit
1381	eSCI: LIN Wakeup flag set after aborted LIN frame transmission
1221	eSCI: LIN bit error indicated at start of transmission after LIN reset

e3521: DECFIL: Soft reset failures at the end of filtering

Errata type: Errata

Description: If a software reset of a decimation filter is made exactly at the time it finishes filtering, several registers reset for one clock, but have their values updated by the filtering on the next clock,

including (but not limited to) the integrator current value register DECFIL_CINTVAL and

the tap registers DECFILTER_TAPn.

Workaround: Before making the soft reset write (DECFIL_MCR bit SRES=1), perform the procedure below:

1- disable filter inputs, writing DECFIL_MCR bit IDIS = 1.

2- read the register DECFIL_MSR and repeat the read until the bit BSY is 0.

3- repeat the loop of step 2; this is necessary to cover the case when a sample is left in the input buffer.

e6026: DSPI: Incorrect SPI Frame Generated in Combined Serial Interface Configuration

Errata type: Errata

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.

e7352: DSPI: reserved bits in slave CTAR are writable

Errata type: Errata

Description: When the Deserial/Serial Peripheral Interface (DSPI) module is operating in slave mode (the Master [MSTR] bit of the DSPI Module Configuration Register [DSPIx_MCR] is cleared), bits 10 to 31 (31 = least significant bit) of the Clock and Transfer Attributes Registers (DSPIx_CTARx) should be read only (and always read 0). However, these bits are writable, but setting any of these bits to a 1 does not change the operation of the module.

Workaround: There are two possible workarounds.

Workaround 1: Always write zeros to the reserved bits of the DSPIx_CTARn_SLAVE (when operating in slave mode).

Workaround 2: Mask the reserved bits of DSPIx_CTARn_SLAVE when reading the register in slave mode.

e3378: EQADC: Pull devices on differential pins may be enabled for a short period of time during and just after POR

Errata type: Errata

Description: The programmable pull devices (up and down) on the analog differential inputs of the eQADC may randomly be enabled during the internal Power On Reset (POR) and until the 1st clock edge propagates through the device. After the first clock edge, the pull resistors will be disabled until software enables them.

Workaround: Protect any external devices connected to the differential analog inputs. The worst case condition is with a 1.4K ohm resistor to VDDA (5K pull-up enabled) or VSSA (5K pull-down enabled). This may also cause temporary additional current requirements on the VDDA supply of each eQADC module, up to 15 mA on each eQADC if both the pull up and pull down resistors are enabled simultaneously on all of the differential analog pins.

e5642: ETPU2: Limitations of forced instructions executed via the debug interface

Errata type: Information

Description: The following limitations apply to forced instructions executed through the Nexus debug interface on the Enhanced Time Processing Unit (ETPU):

1- When a branch or dispatch call instruction with the pipeline flush enabled (field FLS=0) is forced (through the debug port), the Return Address Register (RAR) is updated with the current program counter (PC) value, instead of PC value + 1.

2- The Channel Interrupt and Data Transfer Requests (CIRC) instruction field is not operational.

Workaround: Workaround for limitation #1 (branch or dispatch call instruction):

Increment the PC value stored in the RAR by executing a forced Arithmetic Logic Unit (ALU) instruction after the execution of the branch or dispatch call instruction.

Workaround for limitation #2 (CIRC):

To force an interrupt or DMA request from the debugger:

1- Program a Shared Code Memory (SCM) location with an instruction that issues the interrupt and/or DMA request. Note: Save the original value at the SCM location.

2- Save the address of the next instruction to be executed.

3- Force a jump with flush to the instruction position.

4- Single-step the execution.

5- Restore the saved value to the SCM location (saved in step 1).

6- Force a jump with flush to the address of the next instruction to be executed (saved in step 2).

NOTE: This workaround cannot be executed when the eTPU is in HALT_IDLE state.

e2740: ETPU2: Watchdog Status Register (WDSR) may fail to update on channel timeout

Errata type: Errata

Description: The Watchdog Status Register (WDSR) contains a single watchdog status bit for each of the 32 eTPU channels per engine. When this bit is set, it indicates that the corresponding channel encountered a watchdog timeout and was aborted. Under certain conditions the corresponding bit is not set due to a watchdog timeout, and therefore no indication is available as to which channel timed out. However, the global exception is indicated correctly on a per engine basis, and the correct exception is issued to the interrupt controller and may be serviced.

Workaround: The application software should treat any watchdog event as a global eTPU exception and handle it in the eTPU global exception handler. Additionally, during the global exception handler the application should check the WDSR and clear any bits that may be set by writing '1' to that bit.

e5640: ETPU2: Watchdog timeout may fail in busy length mode

Errata type: Errata

Description: When the Enhanced Time Processing Unit (eTPU) watchdog is programmed for busy length mode (eTPU Watchdog Timer Register (ETPU_WDTR) Watchdog Mode field (WDM) = 3), a watchdog timeout will not be detected if all of the conditions below are met:

1- The watchdog timeout occurs at the time slot transition, at the first instruction of a thread, or at the thread gap. (a thread gap is a 1 microcycle period between threads that service the same channel).

2- The thread has only one instruction.

3- The eTPU goes idle right after the timed-out thread, or after consecutive single-instruction threads.

Workaround: Insert a NOP instruction in threads which have only one instruction.

e3114: FLASH: Erroneous update of the ADR register in case of multiple ECC errors

Description: An erroneous update of the Address register (ADR) occurs whenever there is a sequence of 3 or more events affecting the ADR (ECC single or double bit errors or RWW error) and both the following conditions apply:

- The priorities are ordered in such a way that only the first event should update ADR.

- The last event although it does not update ADR sets the Read While Write Event Error (RWE) or the ECC Data Correction (EDC) in the Module Configuration Register (MCR).

For this case the ADR is wrongly updated with the address related to one of the intervening events.

Example - If a sequence of two double-bit ECC errors is followed by a single-bit correction without clearing the ECC Event Error flag (EER) in the MCR, then the value found in ADR after the single-bit correction event is the one related to the second double-bit error (instead of the first one, as specified)

Workaround: Always process Flash ECC errors as soon as they are detected.

Clear MCR[RWE] at the end of each flash operation (Program, Erase, Array Integrity Check, etc...).

e3196: FLASH: PFCR3 is not directly writable

Description: The Flash Configuration Register 3 (PFCR3) that can control the prefetching settings (Data Prefetch Enable [DPFEN], Instruction Prefetch Enable [IPFEN], Prefetch Limit [PFLIM], and Buffer Enable [BFEN]) of the Bank 1 (array 1 and array 2) flash modules is not directly writable. These settings are enabled by setting the Global Configuration Enable bit in the Flash Bus Interface Unit Control register (BIUCR).

Workaround: Set the GCE bit (BIUCR[GCE=1) to allow the Bank 0, Array 0 prefetch settings to also control bank 1 (Array 1 and Array 2); or program a default value for the PFCR3 register that gets loaded into the register at reset into the Flash Shadow block at address 0x00FF_FE08.

e2379: FMPLL: Loss-of-clock detection may cause unexpected reset

Errata type: Errata

Description: An unexpected Loss-Of-Clock (LOC) event may occur in the following scenario:

1. The FMPLL is initially powered down in bypass mode.

2. The FMPLL is then powered on (still in bypass mode).

3. The LOCK bit of the SYNSR register is polled to determine when the FMPLL is ready.

4. After the LOCK flag becomes set, the FMPLL is switched to normal mode.

5. Loss-of-clock detection is enabled by setting the LOCEN bit of the Enhanced Synthesizer Control Register 2 (ESYNCR2), either before or immediately after switching to normal mode.

The unexpected LOC event will activate the backup clock switching feature, causing the reference clock to be selected as the system clock. If LOC reset was also enabled by setting the LOCRE bit in the ESYNCR2 register, a system reset will occur.

The reason for the unexpected LOC event is that the time it takes for the Clock Quality Monitor (CQM) to detect a valid FMPLL clock is typically larger than the time it takes for the FMPLL to lock. Polling the LOC flag does not help because (the way it is defined) it does not flag LOC in bypass mode.

This issue only occurs when the FMPLL is turned off and then on again without going through a reset cycle. Immediately following reset, the issue can not occur because the CQM keeps the part in reset until it detects a valid crystal clock with plenty of time to detect a valid FMPLL clock.

Workaround: Any time the FMPLL is powered down, wait for 600us before activating the loss-of-clock function.

If the intent is just to re-program the FMPLL, it is not required to turn it off. FMPLL settings can be changed on the fly, and then the CQM will never indicate loss-of-clock.

e5498: Flash: Prefetch during program/erase operation causes system bus stop

Errata type: Errata

Description: While performing a program/erase sequence on one flash bank, prefetches from the other flash bank may cause the system bus to stop.

Workaround: Before initiating any flash program/erase sequence, clear flash the flash prefetch buffers by clearing the PFLASH Line Read Buffer Enable bit in the Flash Bus Interface Unit Configuration Register (FLASH_BIUCR1[BFEN]). After the program/erase sequence is complete, software can re-enable the prefetch buffers by setting FLASH_BIUCR[BFEN].

e7322: FlexCAN: Bus Off Interrupt bit is erroneously asserted when soft reset is performed while FlexCAN is in Bus Off state

Errata type: Errata

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).

e3407: FlexCAN: CAN Transmitter Stall in case of no Remote Frame in response to Tx packet with RTR=1

Errata type: Errata

Description: FlexCAN does not transmit an expected message when the same node detects an incoming Remote Request message asking for any remote answer.

The issue happens when two specific conditions occur:

1) The Message Buffer (MB) configured for remote answer (with code "a") is the last MB. The last MB is specified by Maximum MB field in the Module Configuration Register (MCR[MAXMB]).

2) The incoming Remote Request message does not match its ID against the last MB ID.

While an incoming Remote Request message is being received, the FlexCAN also scans the transmit (Tx) MBs to select the one with the higher priority for the next bus arbitration. It is expected that by the Intermission field it ends up with a selected candidate (winner). The coincidence of conditions (1) and (2) above creates an internal corner case that cancels the Tx winner and therefore no message will be selected for transmission in the next frame. This gives the appearance that the FlexCAN transmitter is stalled or "stops transmitting".

The problem can be detectable only if the message traffic ceases and the CAN bus enters into Idle state after the described sequence of events.

There is NO ISSUE if any of the conditions below holds:

a) The incoming message matches the remote answer MB with code "a".

b) The MB configured as remote answer with code "a" is not the last one.

c) Any MB (despite of being Tx or Rx) is reconfigured (by writing its CS field) just after the Intermission field.

d) A new incoming message sent by any external node starts just after the Intermission field.

Workaround: Do not configure the last MB as a Remote Answer (with code "a").

e5909: MPC563xM/SPC563M: MIDR MASKNUM field is set to 0x22

Errata type: Errata

Description: The mask revision field (MASKNUM[Major, Minor]) of the MCU Identification Register is 0b0010_0010 (0x22).

Workaround: Expect that the MASKNUM fields of the MIDR register will change in the future.

e7590: MPC563xM: Incorrect JTAG ID[MIC] and MIDR[S_F] register values

Errata type: Errata

Description: The Manufacturer's Identification Code (MIC) in the JTAG Device Identification Register and the S_F (manufacturer) bit of the System Integration Unit (SIU) Microcontroller Identification Register 2 (SIU_MIDR2) register may indicate either Freescale or ST randomly on same device. Four bits of the JTAG ID (bits 2, 3, 4, and 6) which are part of the MIC field (bits 11:1 of the JTAG ID) may read as either a 1 or a 0 on each read of the register. Likewise, the S_F bit of the (SIU) Microcontroller Identification Register 2 (SIU_MIDR2) register may indicate either Freescale (0b0) or ST (0b1).

Workaround: Expect that the MSB of the MIDR2 could indicate either Freescale or ST as the manufacturer of the MCU. In addition, expect that 4 of the MIC bits in the JTAG ID will read 0b000_00?0_???0 (the ? bits could be either a 0b0 or a 0b1). All other bits of the JTAG ID and the MIDR2 register will always read correct values. To differentiate between Freescale or ST manufactured material, read the marking on the top of the package.

e3205: NEXUS: EVTI not functional on QFP176 and BGA208 packages

Description: Event In (EVTI) is an input that is read on the negation of TRST (or JCOMP) to enable (if asserted) or disable (if deasserted) the Nexus Debug port.

After reset, EVTI is an input which, when asserted, will initiate one of two events based on the EIC (EVTI Control) bits in the DC1 (Development Control 1) Register (if the Nexus Class 2+ module is enabled at reset):

1) Program Trace and Data Trace synchronization messages (provided Program Trace and EIC = 0b00).

2) Debug request to e200z335 Nexus Class 1 module (provided EIC = 0b01 and this feature is implemented).

Workaround: 1) Do not expect Program Trace Sync messages after EVTI assertion. Other condition for the sync messaging are not impacted.

2) Do not use EVTI to request the CPU to enter the debug state. Other requests are functional.

In case EVTI functionality is needed, CSP496 package can also be used to emulate the 176QFP or the BGA208 packages.

e6726: NPC: MCKO clock may be gated one clock period early when MCKO frequency is programmed as SYS_CLK/8.and gating is enabled

Errata type: Errata

Description: The Nexus auxiliary message clock (MCKO) may be gated one clock period early when the MCKO frequency is programmed as SYS_CLK/8 in the Nexus Port Controller Port Configuration Register (NPC_PCR[MCKO_DIV]=111) and the MCKO gating function is enabled (NPC_PCR[MCKO_GT]=1). In this case, the last MCKO received by the tool prior to the gating will correspond to the END_MESSAGE state. The tool will not receive an MCKO to indicate the transition to the IDLE state, even though the NPC will transition to the IDLE state internally. Upon re-enabling of MCKO, the first MCKO edge will drive the Message Start/End Output (MSEO=11) and move the tool's state to IDLE.

Workaround: Expect to receive the MCKO edge corresponding to the IDLE state upon re-enabling of MCKO after MCKO has been gated.

e3425: PMC: 5V VDDREG POR De-assertion Max Level 4.2V

Errata type: Errata

Description: 5V Voltage regulator input (VDDREG) power on reset (POR) de-assertion maximum specification level is now 4.2V. Previously, the maximum specification level was 4.005V.

Workaround: Expect that 4.2V is required on the VDDREG input before the device will exit from a power on reset. The POR levels for VDDSYN and VDD in the data sheet are unchanged.

e3221: PMC: SRAM standby power low voltage detect circuit is not accurate

Description: The power management controller (PMC) SRAM standby voltage low power detect circuit cannot reliably detect the brown-out condition if the standby supply is below 1.0 volts. The Status Register Brown Out Flag (PMC.SR[LVFSTBY]) bit may not be set during a brownout condition of the SRAM standby voltage or may be set even though no data has been lost.

Workaround: The application software should not rely on the PMC.SR[LVFSTBY] bit to detect corrupted SRAM values.

e2338: Pad Ring: Leakage if VDDE is greater than VDD33

Errata type: Errata

Description: If the VDDEx supplies (provided by an external supply) are greater than the VDD33 supplies (provided by the internal regulator), leakage current can occur through all pins powered by VDDEx from the VDDEx supply on the pad output driver through the pad towards ground. The highest leakage current occurs at high temperatures and is exponentially proportional to the VDDEx-VDD33 differential. Worst case leakage, per grounded pad at 150C is 29uA with a 200mV differential, and 590uA with a 400mV differential in the VDDEx-VDD33.

Any I/O configured as an input with the weak pull down enabled will rise towards VDDE level as the VDDE-VDD33 voltage differential increases (as the leakage current exceeds the weak pull-down capability). The reset state of most Nexus pads is pull-down, so this would not be guaranteed. EVTI is pulled up internally during and after RESET. EVTO must be pulled low externally for Auto-baud rate detection. I/O pads configured as outputs driving LOW will remain below VOL level but will consume the leakage current through the pad driver. External logic driving pads configured as inputs will have to sink this leakage current when driving LOW.

Workaround: Maintain a VDDE-VDD33 voltage difference below 200mV. If VDDE is greater than 3.45V, the PMC_TRIMR[VDD33TRIM] for the internal regulator can be increased to 4 steps above typical (0b1011) to increase VDD33 default voltage by a nominal value of 120mV.

e3377: Pad Ring:Nexus pins may drive an unknown value immediately after power up but before the 1st clock edge

Errata type: Errata

Description: The Nexus Output pins (Message Data outputs 0:15 [MDO] and Message Start/End outputs 0:1 [MSEO]) may drive an unknown value (high or low) immediately after power up but before the 1st clock edge propagates through the device (instead of being weakly pulled low). This may cause high currents if the pins are tied directly to a supply/ground or any low resistance driver (when used as a general purpose input [GPI] in the application).

Workaround: 1. Do not tie the Nexus output pins directly to ground or a power supply.

2. If these pins are used as GPI, limit the current to the ability of the regulator supply to guarantee correct start up of the power supply. Each pin may draw upwards of 150mA.

If not used, the pins may be left unconnected.

e1421: SWT: switching SWT to system clock has very small chance of causing the SWT to enter an indeterminate state

Errata type: Errata

Description: The reset value for the clock source of the Software Watchdog Timer module (SWT) is the oscillator clock. If the clock source is switched to the system clock by clearing Clock Selection bit in the SWT Module Control Register (SWT_MCR[CSL]=0), then the SWT has a very small chance of entering an indeterminate state.

Workaround: Only use the oscillator clock as the SWT clock source.

e4480: eQADC: Differential conversions with 4x gain may halt command processing

Errata type: Errata

Description: If the four times amplifier is enabled for a differential analog-to-digital conversion in the Enhanced Queued Analog to Digital Converter (eQADC) and the ADC clock prescaler is set to divide by 12 or greater, then the ADC will stop processing commands if a conversion command is executed immediately after a differential, gain 4x conversion.

Workaround: 1) Do not use a prescaler divide factor greater than or equal to 12 (11 can be used on devices that support odd prescalers).

2) Insert a dummy write command to any internal ADC register after every 4x conversion command.

Note 1: If the command FIFO preemption feature is used and it is possible to preempt a FIFO which contains the 4x conversion + dummy write workaround, then the preempting command FIFO must be loaded FIRST with a dummy write command and then the desired preempting conversion command in order to avoid the possibility of following a 4x conversion command with another conversion command in the same ADC.

Note 2: The level sensitive triggers (when in Low/High Level Gated External Trigger, Single/ Continuous Scan modes) can interrupt the command sequence at any point in time, potentially breaking the safe sequence 4x conversion command -> dummy write command.

Note 3: When using an odd prescaler (ADCx_CLK_ODD = 1), the duty cycle setting (ADCxCLK_DTY) must be kept at the default setting of 0.

e5086: eQADC: unexpected result may be pushed when Immediate Conversion Command is enabled

Errata type: Errata

Description: In the enhanced Queued Analog to Digital Converter (eQADC), when the Immediate Conversion Command is enabled (ICEAn=1) in the eQADC_MCR (Module Configuration Register), if a conversion from Command First-In-First Out (CFIFO0, conv0) is requested concurrently with the end-of-conversion from another, lower priority conversion (convx), the result of the convx may be lost or duplicated causing an unexpected number of results in the FIFO (too few or too many).

Workaround: Workaround 1: Do not use the abort feature (ICEAn=0).

Workaround 2: Arrange the timing of the CFIFO0 trigger such that it does not assert the trigger at the end of another, lower priority conversion.

Workaround 3: Detect the extra or missing conversion result by checking the EQADC_CFTCRx (EQADC CFIFO Transfer Counter Register x). This register records how many commands were issued, so it can be used to check that the expected number of results have been received.

e1297: eSCI : reads of the SCI Data Register, which clears the RDRF flag, may cause loss of frame if read occurs during reception of the STOP bit

Errata type: Errata

Description: A received SCI frame is not written into the SCI Data Registers and the Overrun (OR) flag is not set in the SCI Status Register 1 (SCISR1), if:

1.) The eSCI has received the last data bit of an SCI frame n

2.) and the Receive Data Register Full (RDRF) flag is still set in the SCISR1 after the reception of SCI frame n-1

3.) and during the reception of the STOP bit of frame n the host reads the SCI Data Registers, and clears the RDRF flag

In this case the RDRF flag is erroneously set again by the controller instead of the OR flag. Thus, the host reads the data of frame n-1 a second time, and the data of frame n is lost.

Workaround: The application should ensure that the data of the foregoing frame is read out from the SCI Data Registers before the last data bit of the actual frame is received.

e1381: eSCI: LIN Wakeup flag set after aborted LIN frame transmission

Errata type: Errata

Description: If the eSCI module is transmitting a LIN frame and the application sets and clears the LIN Finite State Machine Resync bit in the LIN Control Register 1 (eSCI_LCR1[LRES]) to abort the transmission, the LIN Wakeup Receive Flag in the LIN Status Register may be set (LWAKE=1).

Workaround: If the application has triggered LIN Protocol Engine Reset via the eSCI_LCR1[LRES], it should wait for the duration of a frame and clear the eSCI_IFSR2[LWAKE] flag before waiting for a wakeup.

e1221: eSCI: LIN bit error indicated at start of transmission after LIN reset

Errata type: Errata

Description: If the eSCI module is in LIN mode and is transmitting a LIN frame, and the application sets and subsequently clears the LIN reset bit (LRES) in the LIN Control register 1 (ESCI_LCR1), the next LIN frame transmission might incorrectly signal the occurrence of bit errors (ESCI_IFSR1[BERR]) and frame error (ESCI_IFSR1[FE]), and the transmitted frame might be incorrect.

Workaround: There is no generic work around. The implementation of a suitable workaround is highly dependent on the application and a workaround may not be possible for all applications.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

freescale™