Mask Set Errata for Mask 0M22Y

Introduction

This report applies to mask 0M22Y for these products:

- MPC5602P

Errata ID	Errata Title
2973	ADC ABORT CHAIN bit
2971	ADC ABORT bit (single conversion)
3069	ADC: ADC_DMAE[DCLR] set to 1 clears the DMA request incorrectly
2997	ADC: Injected conversion not executed during scan mode.
2972	ADC: Last conversion in chain not aborted
2894	ADC: Minimum sampling time for ADC at 32MHz
2897	ADC: Offset Cancellation Not Required
3010	ADC: conversion chain failing after ABORT chain
3248	ADC:Abort request during last sampling cycle corrupts the data register of next channel conversion
2963	CMU XOSC Monitoring cannot be guaranteed when RCDIV>0 and FOSC is less than 1.5*Frc/2^rcdiv
3442	CMU monitor: FXOSC/FIRC and FMPLL/FIRC relation
575	DSPI: Changing CTARs between frames in continuous PCS mode causes error
1103	DSPI: PCS Continuous Selection Format limitation
1082	DSPI: set up enough ASC time when MTFE=1 and CPHA=1
3110	Debugging functionality could be lost when unsecuring a secured device.
3164	FCU: Timeout feature doesn't work correctly.
3324	FIRC - FIRC_CTL[TRIM] does not display correct trim value after reset
2981	FMPLL: Do not poll flag FMPLL_CR[PLL_FAIL]
2883	FMPLL: FMPLL_CR[UNLOCK_ONCE] wrongly set
2656	FlexCAN: Abort request blocks the CODE field
3407	FlexCAN: CAN Transmitter Stall in case of no Remote Frame in response to Tx packet with RTR=1
2360	FlexCAN: Global Masks misalignment

Table continues on the next page...

Errata ID	Errata Title
2685	FlexCAN: Module Disable Mode functionality not described correctly
3511	FlexPWM : Incorrect PWM operation when IPOL is set.
3257	FlexPWM: Configuring the count value to set PWMA/PWMB first low and then high in the same cycle the output signals are low.
3335	FlexPWM: Incorrect PWM operation when mixing DMA and non-DMA controlled channels
3310	FlexPWM: PWM signals are improperly synced when using Master Sync
3028	LINFlex: BDRL/BDRM cannot be accessed as byte or half-word
3021	LINFlex: Unexpected LIN timeout in slave mode
3264	MCM: MRSR does not report Power On Reset event
2999	MC_CGM and MC_PCU: A data storage exception is not generated on an access to MC_CGM or MC_PCU when the respective peripheral is disabled at MC_ME
3220	MC_CGM: system clock may stop in case target clock source dies during clock switching
3269	MC_ME: Peripheral clocks may get incorrectly disabled or enabled after entering debug mode
3584	MC_ME: Possibility of Machine Check on Low-Power Mode Exit
3583	MC_RGM: A non-monotonic ramp on the VDD_HV_REG supply can cause the RGM module to clear all flags in the DES register.
2958	MC_RGM: Clearing a flag at RGM_DES or RGM_FES register may be prevented by a reset
3060	MC_RGM: SAFE mode exit may be possible even though condition causing the SAFE mode request has not been cleared
3377	Pad Ring:Nexus pins may drive an unknown value immediately after power up but before the 1st clock edge
3262	Register Protection on full CMU_CSR
3263	Serial Boot and Censorship: flash read access
3256	eTimer: Configuring an eTimer counter channel in GATED-COUNT mode or in SIGNED-COUNT mode may cause a possible incorrect counting of 1 tick.

e2973: ADC ABORT CHAIN bit

Errata type: Errata
Description: If user aborts a chain of ADC conversions, the current conversion appears as aborted but the relative data register is however updated.

Workaround: None

e2971: ADC ABORT bit (single conversion)

Errata type: Errata
Description: If the User starts one single ADC conversion and, after, starts a chain of conversions, when the User tries to abort one of the chained conversions, the abort command aborts the chain instead of a single conversion of the chain.

Workaround: - A minimum of two conversions must be programmed.

- A dummy conversion must be started before or after a single conversion.

e3069: ADC: ADC_DMAE[DCLR] set to 1 clears the DMA request incorrectly

Errata type: n/a
Description: When ADC_DMAE[DCLR] is set the DMA request should be cleared only after the data registers are read. However for this case the DMA request is automatically cleared and will not be recognised by the eDMA.

Workaround: None

e2997: ADC: Injected conversion not executed during scan mode.

Errata type: n/a
Description: When ADC is converting a chain in scan mode -configured using NSTART bit in non-CTU mode

operation; and a injected conversion arrives -triggered by software with JSTART bit or by hardware from eTimer_1 channel 5 (internal connection); the ADC gets stuck in the sampling phase (the triggered conversion is not executed and the chain is not restarted).

Workaround: None

e2972: ADC: Last conversion in chain not aborted

Errata type: Errata
Description: If the user aborts the last ADC conversion of a chain the conversion appears to be aborted but the relevant data register is updated with the conversion data.

Workaround: None

e2894: ADC: Minimum sampling time for ADC at 32MHz

Errata type: n/a
Description: When ADC is running at 32MHz the minimum sampling time of 135ns specified is not met.

Workaround: At 32MHZ the minimum sampling time must be at least 180ns.

e2897: ADC: Offset Cancellation Not Required

Errata type: Errata
Description: The offset cancellation mechanism does not improve the ADC performance as the intrinsic ADC precision is better than the offset cancellation resolution.

Workaround: Do not use the offset cancellation feature as it will not improve ADC precision.

e3010: ADC: conversion chain failing after ABORT chain

Errata type: n/a

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.

e3248: ADC:Abort request during last sampling cycle corrupts the data register of next channel conversion

Errata type: n/a

Description: If the abort pulse is valid in the last cycle of the SAMPLE phase, the current channel is correctly aborted but the data register (CDR[0..15]) of the next channel conversion shows an invalid value.

Workaround: None

e2963: CMU XOSC Monitoring cannot be guaranteed when RCDIV>0 and FOSC is less than 1.5*Frc/2^rcdiv

Errata type: Errata

Description: A False OLRI (Oscillator frequency less than RC frequency) event may be generated when FOSC is less than 1.5*(FRC / 2^RCDIV) and RCDIV>0, and not FRC/ 2^RCDIV as described in the Reference Manual).

Correct crystal clock monitoring is guaranteed when FOSC is strictly above 1.5*(FRC / 2^RCDIV).

Workaround: There are 2 workarounds available :

1. Keep RCDIV = 0

2. When RCDIV > 0, ensure FOSC is greater than FRC/2^(RCDIV -1) to avoid false OLRI (CMU external oscillator failure) event.

e3442: CMU monitor: FXOSC/FIRC and FMPLL/FIRC relation

Errata type: Errata

Description: Functional CMU monitoring can only be guaranteed when the following conditions are met:

- FXOSC frequency must be greater than (FIRC / 2^RCDIV) + 0.5MHz in order to guarantee correct FXOSC monitoring

- FMPLL frequency must be greater than (FIRC / 4) + 0.5MHz in order to guarantee correct FMPLL monitoring

Workaround: Refer to description

e575: DSPI: Changing CTARs between frames in continuous PCS mode causes error

Errata type: Errata

Description: Erroneous data could be transmitted if multiple Clock and Transfer Attribute Registers (CTAR) are used while using the Continuous Peripheral Chip Select mode (DSPIx_PUSHR[CONT=1]). The conditions that can generate an error are:

1) If DSPIx_CTARn[CPHA]=1 and DSPIx_MCR[CONT_SCKE = 0] and DSPIx_CTARn[CPOL, CPHA, PCSSCK or PBR] change between frames.

2) If DSPIx_CTARn[CPHA]=0 or DSPIx_MCR[CONT_SCKE = 1] and any bit field of DSPIx_CTARn changes between frames except DSPIx_CTARn[PBR].

Workaround: When generating DSPI bit frames in continuous PCS mode, adhere to the aforementioned conditions when changing DSPIx_CTARn bit fields between frames.

e1103: DSPI: PCS Continuous Selection Format limitation

Errata type: Errata

Description: When the DSPI module has more than one entry in the TX FIFO and only one entry is written and that entry has the CONT bit set, and continuous SCK clock selected the PCS levels may change between transfer complete and write of the next data to the DSPI_PUSHR register.

For example:

If the CONT bit is set with the first PUSHR write, the PCS de-asserts after the transfer because the configuration data for the next frame has already been fetched from the next (empty) fifo entry. This behavior continues till the buffer is filled once and all CONT bits are one.

Workaround: To insure PCS stability during data transmission in Continious Selection Format and Continious SCK clock enabled make sure that the data with reset CONT bit is written to DSPI_PUSHR register before previous data sub-frame (with CONT bit set) transfer is over.

e1082: DSPI: set up enough ASC time when MTFE=1 and CPHA=1

Errata type: Information

Description: When the DSPI is being used in the Modified Transfer Format mode (DSPI_MCR[MTFE]=1) with the clock phase set for Data changing on the leading edge of the clock and captured on the following edge in the DSPI Clock and Transfer Attributes Register (DSPI_CTARn[CPHA]=1), if the After SCK delay scaler (ASC) time is set to less than 1/2 SCK clock period the DSPI may not complete the transaction - the TCF flag will not be set, serial data will not received, and last transmitted bit can be truncated.

Workaround: If the Modified Transfer Format mode is required DSPI_MCR[MTFE]=1 with the clock phase set for serial data changing on the leading edge of the clock and captured on the following edge in the SCK clock (Transfer Attributes Register (DSPI_CTARn[CPHA]=1) make sure that the ASC time is set to be longer than half SCK clock period.

e3110: Debugging functionality could be lost when unsecuring a secured device.

Errata type: n/a

Description: Providing the backdoor password via JTAG or via serial boot would unsecure the device, but on some devices may leave the Nexus interface and potentially the CPUs in an undetermined state.

Normal operation without a debugger is unaffected, debugging unsecured devices is also unaffected.

Workaround: A second connection attempt may be successful.

Boot in serial mode (using the Flash password), then execute code which unsecures the device.(The JTAG interface needs to be inactive while the unsecure happens.)

Implement a separate backdoor in application software. Once the software detects the custom backdoor sequence it can unlock the device via Flash write.

Leave device unsecured for debugging.

e3164: FCU: Timeout feature doesn't work correctly.

Errata type: n/a

Description: A fault occurs, timeout for this fault is enabled and the fault is not recovered automatically before the timeout :- In this case device will go to ALARM state and waits for timeout, after timeout has elapsed it enters to FAULT state. However, if another fault occurs with timeout enabled the FCU will go directly to FAULT state without waiting for timeout to be elapsed.

If it is desired that FCU will go again to ALARM state for the second fault, a Watchdog reset needs to be asserted after first fault is detected.

Workaround: None

e3324: FIRC - FIRC_CTL[TRIM] does not display correct trim value after reset

Errata type: n/a

Description: The FIRC is trimmed during reset using a factory programmed value stored in flash. However after reset the trim value is not copied to FIRCRC_CTL[TRIM] as one would expect. Any read of FIRCRC_CTL[TRIM] will read 0 and in all likelihood this will not be the factory programmed value. Therefore any read-modify-write on the 32-bit register will set the FIRC to a trim value of 0 and not the factory programmed value.

Workaround: As the lower 16-bits of FIRC_CTL register only contain the TRIM field it is recommended that if the user wishes to program any other field they should only access the upper 16-bits of this register.

If the user wishes to calibrate the FIRC this should be performed using the CMU.

e2981: FMPLL: Do not poll flag FMPLL_CR[PLL_FAIL]

Errata type: Errata

Description: For the case when the FMPLL is indicating loss of lock the flag FMPLL_CR[PLL_FAIL] is unpredictable.

Workaround: To avoid reading an incorrect value of FMPLL_CR[PLL_FAIL] only read this flag inside the FMPLL Interrupt Service Routine (ISR). The ISR indicates the flag has been set correctly and at this point the flag can be cleared. Do not poll flag FMPLL_CR[PLL_FAIL] at any other point in the application software.

e2883: FMPLL: FMPLL_CR[UNLOCK_ONCE] wrongly set

Errata type: n/a
Description: If the FMPLL is locked and a functional reset occurs, FMPLL_CR[UNLOCK_ONCE] is automatically set even when the FMPLL has not lost lock.

Workaround: Do not use the FMPLL_CR[UNLOCK_ONCE] when a functional reset occurs.

e2656: FlexCAN: Abort request blocks the CODE field

Errata type: Errata
Description: An Abort request to a transmit Message Buffer (TxMB) can block any write operation into its CODE field. Therefore, the TxMB cannot be aborted or deactivated until it completes a valid transmission (by winning the CAN bus arbitration and transmitting the contents of the TxMB).

Workaround: Instead of aborting the transmission, use deactivation instead.

Note that there is a chance the the deactivated TxMB can be transmitted without setting IFLAG and updating the CODE field if it is deactivated.

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").

e2360: FlexCAN: Global Masks misalignment

Errata type: Errata
Description: Convention: MSB=0.

During CAN messages reception by FlexCAN, the RXGMASK (Rx Global Mask) is used as acceptance mask for most of the Rx Message Buffers (MB). When the FIFO Enable bit in the FlexCAN Module Configuration Register (CANx_MCR[FEN], bit 2) is set, the RXGMASK also applies to most of the elements of the ID filter table. However there is a misalignment between the position of the ID field in the Rx MB and in RXIDA, RXIDB and RXIDC fields of the ID Tables. In fact RXIDA filter in the ID Tables is shifted one bit to the left from Rx MBs ID position as shown below:

Rx MB ID = bits 3-31 of ID word corresponding to message ID bits 0-28

RXIDA = bits 2-30 of ID Table corresponding to message ID bits 0-28

Note that the mask bits one-to-one correspondence occurs with the filters bits, not with the incoming message ID bits.

This leads the RXGMASK to affect Rx MB and Rx FIFO filtering in different ways.

For example, if the user intends to mask out the bit 24 of the ID filter of Message Buffers then the RXGMASK will be configured as 0xffff_ffef. As result, bit 24 of the ID field of the incoming message will be ignored during filtering process for Message Buffers. This very same configuration of RXGMASK would lead bit 24 of RXIDA to be "don't care" and thus bit 25 of the ID field of the incoming message would be ignored during filtering process for Rx FIFO.

Similarly, both RXIDB and RXIDC filters have multiple misalignments with regards to position of ID field in Rx MBs, which can lead to erroneous masking during filtering process for either Rx FIFO or MBs.

RX14MASK (Rx 14 Mask) and RX15MASK (Rx 15 Mask) have the same structure as the RXGMASK. This includes the misalignment problem between the position of the ID field in the Rx MBs and in RXIDA, RXIDB and RXIDC fields of the ID Tables.

Workaround: Therefore it is recommended that one of the following actions be taken to avoid problems:

*) Do not enable the RxFIFO. If CANx_MCR[FEN]=0 then the Rx FIFO is disabled and thus the masks RXGMASK, RX14MASK and RX15MASK do not affect it.

*) Enable Rx Individual Mask Registers. If the Backwards Compatibility Configuration bit in the FlexCAN Module Configuration Register (CANx_MCR[BCC], bit 15) is set then the Rx Individual Mask Registers (RXIMR0-63) are enabled and thus the masks RXGMASK, RX14MASK and RX15MASK are not used.

*) Do not use masks RXGMASK, RX14MASK and RX15MASK (i.e. let them in reset value which is 0xffff_ffff) when CANx_MCR[FEN]=1 and CANx_MCR[BCC]=0. In this case, filtering processes for both Rx MBs and Rx FIFO are not affected by those masks.

*) Do not configure any MB as Rx (i.e. let all MBs as either Tx or inactive) when CANx_MCR[FEN]=1 and CANx_MCR[BCC]=0. In this case, the masks RXGMASK, RX14MASK and RX15MASK can be used to affect ID Tables without affecting filtering process for Rx MBs.

e2685: FlexCAN: Module Disable Mode functionality not described correctly

Errata type: Errata

Description: Module Disable Mode functionality is described as the FlexCAN block is directly responsible for shutting down the clocks for both CAN Protocol Interface (CPI) and Message Buffer Management (MBM) sub-modules. In fact, FlexCAN requests this action to an external logic.

Workaround: In FlexCAN documentation chapter:

Section "Modes of Operation", bullet "Module Disable Mode":

Where is written:

"This low power mode is entered when the MDIS bit in the MCR Register is asserted. When disabled, the module shuts down the clocks to the CAN Protocol Interface and Message Buffer Management sub-modules.."

The correct description is:

"This low power mode is entered when the MDIS bit in the MCR Register is asserted by the CPU. When disabled, the module requests to disable the clocks to the CAN Protocol Interface and Message Buffer Management sub-modules."

Section "Modes of Operation Details", Sub-section "Module Disable Mode":

Where is written:

"This low power mode is entered when the MDIS bit in the MCR Register is asserted. If the module is disabled during Freeze Mode, it shuts down the clocks to the CPI and MBM sub-modules, sets the LPM_ACK bit and negates the FRZ_ACK bit.."

The correct description is:

"This low power mode is entered when the MDIS bit in the MCR Register is asserted. If the module is disabled during Freeze Mode, it requests to disable the clocks to the CAN Protocol Interface (CPI) and Message Buffer Management (MBM) sub-modules, sets the LPM_ACK bit and negates the FRZ_ACK bit."

e3511: FlexPWM : Incorrect PWM operation when IPOL is set.

Errata type: Errata

Description: When IPOL bit is set in complementary mode, the source of the PWMi waveform is supposed to be switched from the VAL2 and VAL3 registers to the VAL4 and VAL5 registers. This switch is not happening.

Workaround: Instead of setting IPOL bt, the application can swap the VAL2/3 values with the VAL4/5 values.

e3257: FlexPWM: Configuring the count value to set PWMA/PWMB first low and then high in the same cycle the output signals are low.

Errata type: n/a

Description: The VAL2 and VAL4 registers define the turn-on edge and the VAL3 and VAL5 registers define the turn off edge of the PWMA/PWMB signals respectively. VAL3 cannot be less than VAL2 and VAL5 cannot be less than VAL4. Doing so will cause the PWM signal to turn off at the correct time (VAL3 or VAL5), but it will not turn on at the time defined by VAL2 or VAL4.

This can be an issue during the generation of phase delayed pulses where the PWM signal goes high late in PWM cycle N and remains high across the cycle boundary before going low early in cycle N+1 and goes high again in PWM cycle N+1. This errata will allow that to happen.

VAL3 register must be "greater than or equal to" VAL2 register and VAL5 must be "greater than or equal to" VAL4.

Workaround: None

e3335: FlexPWM: Incorrect PWM operation when mixing DMA and non-DMA controlled channels

Errata type: Errata

Description: When some submodules use DMA to load their VALx registers and other submodules use non-DMA means that means direct writes from the CPU, the LDOK bits for the non-DMA submodules can be incorrectly cleared at the completion of the DMA controlled load cycle. This leads to the non-DMA channels not being properly updated.

Submodules that use DMA to read the input capture registers do not cause a problem for non-DMA submodules.

Workaround: Set the DMA enable bit to 1 also for non-DMA submodules, according to this the DMA will not incorrectly clear the LDOK bit for non-DMA submodules but they will be set to 1 at the end of each DMA cycle. When the CPU has to update the VALx registers of non-DMA submodules, first clear LDOK bit for non-DMA submodules

e3310: FlexPWM: PWM signals are improperly synced when using Master Sync

Errata type: n/a

Description: If Master Sync signal , originated as the Local Sync from sub-module 0, is selected as the counter initialization signal for sub-modules 1-2-3 (slaves), with a prescaler PRSC < 0x2 on PWM clock, the slave sub-module PWM outputs are delayed approx 2 IP clock against sub-module0.

For PRSC > 0x1 the delay on slave sub-modules disappears.

Workaround: If Master Sync signal is requested, use a prescaler value PRSC >=0x2 to synchronize PWM outputs of Sub-module 0 to slave sub-modules PWM outputs, or don't use the sub-module 0 channel A and B.

e3028: LINFlex: BDRL/BDRM cannot be accessed as byte or half-word

Errata type: Errata

Description: LINFlex data buffers (BDRL/BDRM) cannot be accessed as byte or halfword. Accessing BDRL/BDRM in byte/half word mode will lead to incorrect data writing/reading.

Workaround: Access BDRL/BDRM registers as word only.

e3021: LINFlex: Unexpected LIN timeout in slave mode

Errata type: n/a

Description: If the LINFlex is configured in LIN slave mode, an unexpected LIN timeout event (LINESR[OCF]) may occur during LIN Break reception.

Workaround: It is recommended to disable this functionality during LINFlex initialization by clearing LINTCSR[IOT] and LINIER[OCIE] bits, and ignore timeout events.

e3264: MCM: MRSR does not report Power On Reset event

Errata type: n/a

Description: The flag POR of MRSR register stays low after Power On Reset event on the device.

Workaround: Do not use MRSR[POR] to determine Power on reset cause. Use RGM_DES instead.

e2999: MC_CGM and MC_PCU: A data storage exception is not generated on an access to MC_CGM or MC_PCU when the respective peripheral is disabled at MC_ME

Errata type: Errata

Description: If a peripheral with registers mapped to MC_CGM or MC_PCU address spaces is disabled via the MC_ME any read or write accesses to this peripheral will be ignored without producing a data storage exception.

Workaround: For any mode other than a low-power mode do not disable any peripheral that is mapped to MC_CGM or MC_PCU.

e3220: MC_CGM: system clock may stop in case target clock source dies during clock switching

Errata type: n/a

Description: The clock switching is a two step process. The availability of the target clock is first verified. Then the system clock is switched to the new target clock source within two target clock cycles.

For the case when the FXOSC stops during the required two cycles, the switching process may not complete, causing the system clock to stop and prevent further clock switching. This may happen if one of the following cases occurs while the system clock source is switching to FXOSC:

- FXOSC oscillator failure

- SAFE mode request occurs, as this mode will immediately switch OFF the FXOSC (refer to ME_SAFE_MC register configuration)

Workaround: The device is able to recover through any destructive reset event, so typically either the SWT (internal watchdog) will generate a reset and the device will restart properly after reset.

To reduce the probability that this issue occurs in the application, it is recommended to disable SAFE mode transitions when the device is executing a mode transition with the FXOSC as the system clock source in the target mode.

e3269: MC_ME: Peripheral clocks may get incorrectly disabled or enabled after entering debug mode

Errata type: Errata

Description: If ME_RUN_PCx, ME_LP_PCx, ME_PCTLx registers are changed to enable or disable a peripheral, and the device enters debug mode before a subsequent mode transition, the peripheral clock gets enabled or disabled according to the new configuration programmed. Also ME_PSx registers will report incorrect status as the peripheral clock status is not expected to change on debug mode entry.

Workaround: After modifying any of the ME_RUN_PCx, ME_LP_PCx, ME_PCTLx registers, request a mode change and wait for the mode change to be completed before entering debug mode in order to have consistent behaviour on peripheral clock control process and clock status reporting in the ME_PSx registers.

e3584: MC_ME: Possibility of Machine Check on Low-Power Mode Exit

Errata type: Errata

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 machine check due to the flash not being available on RUNx mode re-entry. This will cause either a checkstop reset or machine check interrupt.

Workaround: This issue can be handled in one of the following ways. Workaround #1 configures the application to handle the machine check interrupt in RAM dealing with the problem if it occurs. Workaround #2 configures the MCU to avoid the machine check interrupt.

Workaround #1

The application can be configured to handle the machine check interrupt in RAM; when this occurs, the Mode Entry module can be used to bring up the flash normally and resume execution. Before stop mode entry, ensure the following:

1. Enable the Machine Check interrupt at the core MSR[ME] – this prevents a machine check reset occurring

2. Copy IVOR vector table to RAM

3. Point IVPR to vector table in RAM

4. Implement Machine Check interrupt handler in RAM to power-cycle flash to synchronise status of flash between Mode Entry and flash module

The interrupt handler should perform the following steps:

1. Test Machine Check at LPM exit due to wakeup/interrupt event

2. ME_RUNx_MC[CFLAON] = 0b01 (power-down)

3. Re-enter mode RUNx (x = 0,1,2,3) to power down flash

4. Wait for transition to RUNx mode to complete (ME_GS[S_MTRANS] = 1)

5. ME_RUNx_MC[CFLAON] = 0b11 (normal)

6. Re-enter mode RUNx (x = previous x) to power up flash

7. Wait for transition to RUNx mode to complete (ME_GS[S_MTRANS] = 1)

8. On completion, code execution will return to flash (via se_rfci)

Workaround #2

The application can be configured to avoid the machine check interrupt; low-power mode can be entered from a RAM function and Mode Entry configured to have flash off on return to the current RUNx mode. Flash can then be re-enabled by Mode Entry within the RAM function before returning to execution from flash.

1. Prior to LPM mode entry request branch to code execution in RAM while flash is still in normal mode

2. Set ME_RUNx_MC[CFLAON] = 0b01 (power-down) or 0b10 (low-power) for STOP0/HALT0

3. Set ME_STOP0/HALT0_MC[CFLAON] = ME_RUNx_MC[CFLAON]

4. Enter STOP0/HALT0 mode

5. At wakeup or interrupt from STOP0/HALT0, MCU enters RUNx mode executing from RAM with flash in low-power or power-down as per the ME_RUNx_MC configuration from step 2.

6. After the STOP0/HALT0 request, set ME_RUNx_MC[CFLAON] = 0b11 (normal)

7. Enter RUNx mode

8. Wait for transition to RUNx mode to complete (ME_GS[S_MTRANS] = 0)

9. Return to code execution in flash

e3583: MC_RGM: A non-monotonic ramp on the VDD_HV_REG supply can cause the RGM module to clear all flags in the DES register.

Errata type: Errata

Description: At system power-up a non-monotonic voltage ramp-up or a very slow voltage ramp-up (also known as ‘soft start-up’) can cause incorrect flag setting in the RGM_DES register.

During monotonic power-up, F_POR flag is set when the high voltage regulator supply (VDD_HV_REG) goes above LVD27_VREG low voltage detector threshold and the 1.2V supply (VDD_LV_REGCOR) goes above LVD12_PD0 low voltage detector threshold. Expected behavior POR =1 , LVD27 =0, LVD12=0

During a non-monotonic power-up the VDD_HV_REG may show a non-linearity in the ramp up. When the VDD_HV_REG supply dips below LVD27_VREG threshold, LVD27_VREG low voltage detector is re-fired. If VDD_LV_REGCOR is already above LVD12_PD0 low voltage detector threshold, F_POR flag is reset and F_LVD27_VREG is set. Expected behavior POR=0 , LVD27 =1, LVD12=0

This errata reports behavior when the non-linearity on VDD_HV_REG coincides with the ramp-up of VDD_LV_REGCOR completion and LVD27_VREG is re-fired just after the LVD12_PD0 is released. In this case, neither F_POR flag nor F_LVD27_VREG flag will be set. In this case, application code cannot use the flags to tell if a power-on reset has occurred.

This errata only affects the flag circuit and not a the device initialization. Device initializes correctly under all conditions.

Workaround: Hardware Workaround

Ensure that non-linearity on VDD_HV_REG is < 100mV . This is the hysteresis limit of the device. Board regulator should be chosen accordingly.

Software Workaround

The software workaround need only be applied when neither the F_POR, LVD27 or LVD12 flags are set and involves checking SRAM contents and monitoring for ECC errors during this process. In all cases, an ECC error is assumed to signify a power-on reset (POR).

Two suggestions are made for software workarounds. In both cases, if POR is detected all RAM should be initialized otherwise no power-on condition is detected and it is possible to initialize only needed parts of RAM while preserving required information.

Software workaround #1 :

An area of RAM can be reserved by the compiler into which a KEY, such as 0x3EC1_9678, is written. This area can be checked at boot and if the KEY is incorrect or an ECC error occurs, POR can be assumed and the KEY should be set. Use of a KEY increases detection rate to 31 bits(=<10e-9) or 23bits (= <5.10e-6) instead of 7-bit linked to ECC (=<10e-2)

Software workaround #2 :

When runtime data should be retained and RAM only re-initialized in the case of POR, the CRC module should be used to calculate and store a CRC signature when writing data that can be checked at boot time. If CRC signature is incorrect, POR can be assumed.

e2958: MC_RGM: Clearing a flag at RGM_DES or RGM_FES register may be prevented by a reset

Errata type: Errata

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.

e3060: MC_RGM: SAFE mode exit may be possible even though condition causing the SAFE mode request has not been cleared

Errata type: n/a

Description: A SAFE mode exit should not be possible as long as any condition that caused a SAFE mode entry is still active. However, if the corresponding status flag in the RGM_FES register has been cleared, the SAFE mode exit may incorrectly occur even though the actual condition is still active.

Workaround: Software must clear the SAFE mode request condition at the source before clearing the corresponding RGM_FES flag. This will ensure that the condition is no longer active when the RGM_FES flag is cleared and thus the SAFE mode exit can occur under the correct conditions.

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.

e3262: Register Protection on full CMU_CSR

Errata type: n/a

Description: The register protection on CMU_CSR of CMU0 works only on the full 32 bit, while it should protect only the bits 24-31.

As a consequence, when register protection is active on CMU_CSR the frequency meter cannot be used anymore.

Workaround: In order to perform a frequency meter operation, the register protection of the relevant CMU must be disabled first; this workaround would work only when soft lock is active.

e3263: Serial Boot and Censorship: flash read access

Errata type: n/a

Description: In a secured device, starting with a serial boot, it is possible to read the content of the four flash locations where the RCHW is stored.

For example if the RCHW is stored at address 0x00000000, the reads at address 0x00000000, 0x00000004, 0x00000008 and 0x0000000C will return a correct value.

Any other flash address is not readable.

Workaround: No workaround

e3256: eTimer: Configuring an eTimer counter channel in GATED-COUNT mode or in SIGNED-COUNT mode may cause a possible incorrect counting of 1 tick.

Errata type: n/a

Description: This bug affects eTimer in the following counting modes:

- GATED-COUNT mode, the CNTMODE field is '011' (count rising edges of primary source while secondary input is high);

-SIGNED-COUNT mode, the CNTMODE field is '101' (count primary source rising edges, secondary source specifies direction (up/down)).

Delays in the edge detection circuitry lead to a bug where the rising edge on the primary source is compared to the secondary source value one clock later in time. This means that if there is a rising edge on the primary source followed immediately by the secondary source going high, the eTimer logic could see this as a rising primary edge while the secondary is high even though the secondary input was low at the time of the rising primary edge.

► Note: The counter will also increment if the primary source is already high when the secondary source goes high.

The inputs are sampled by MOTC_CLK and a transition detector finds the rising edge.It is not an asynchronous counter.Also, transitions are limited to one rising or falling edge per clock period. The primary and secondary inputs come into the Timer and are sampled. The primary input is checked for a rising edge which takes another clock period. This extra cycle of delay on only the primary input is the problem.

If the primary source transitions high while the secondary is low, then no counting should occur. If the secondary source transitions high in the clock cycle after the primary source transition, then the extra delay to find the rising edge will confuse the counter and it will think , that the primary rising edge occurred while the secondary was high and will increment the counter.

If the transitions of the primary and secondary sources are more than one clock period apart, then there will be no incorrect counting.

► Possibly, the user will accept that if the edges occur this close together, then its okay to consider it as a countable occurrence, this is application dependant and the responsibility of the user.

Workaround: The source selected as the secondary input to the eTimer channel needs to have an additional clock cycle of delay added to it. This can be achieved by using the input filters.

For the primary source set FILT_PER==1 and FILT_CNT==0.

For the secondary source set FILT_PER==1 and FILT_CNT==1.

This will introduce a 5 clock cycle latency on the primary source and a 6 cycle latency on the secondary source which will properly align the two signals for Gated and Signed count modes.

Note: Ensure that the primary source is low before the secondary source goes high to avoid a false count.

Note: This work-around is only valid when the sources are external an pass through the input filter, internal signals have no work-around.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: MPC5602P_0M22Y
Rev. 05 OCT 2011

freescale semiconductor