Mask Set Errata for Mask 1N10D

This report applies to mask 1N10D for these products:

- MPC5604E

Errata Number	Errata Title
7322	FlexCAN: Bus Off Interrupt bit is erroneously asserted when soft reset is performed while FlexCAN is in Bus Off state
7394	MC_ME: Incorrect mode may be entered on low-power mode exit.
5865	Video Encoder output buffer access is stalled
6239	eTimer: When counter is not enabled, capture flags can be set but capture register values are not updated.
6802	eTimer: Extra input capture events can set unwanted DMA requests
6583	eTimer: Incorrect updating of the Hold Register

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).

e7394: MC_ME: Incorrect mode may be entered on low-power mode exit.

Errata type: Errata

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. */
```

e5865: Video Encoder output buffer access is stalled

Errata type: Errata

Description: The Video Encoder output buffer gets locked and stalls an ongoing Cross-bar system bus access, if a concurrent event, Cross-bar system bus read access to the Video Encoder output together with one of the Video Encoder events: End of Image, Start of Image, or Alarm event occurs. As the Video Encoder output buffer shares the same Cross-bar system bus port as the FLASH memory, flash memory access is also stalled.

Since the access is locked, Fast Ethernet Controller will not be able to read out data from the Video Encoder output buffer causing on ongoing Ethernet transmit being aborted and not finished due to a data underrun.

Workaround: The Video Encoder output buffer lock recovers when Video Encoder does a write access to its output buffer in the even cycle.

Dependency on the Video Encoder mode:

a) Continuous encoding (VE_MODE[AUTOCLR_GO]=0)

The lock resolves with the next Video Encoder write occurring on an even cycle.

All read accesses should be finished before the next Video Encoder event.

After EOI from the JPEG encoder, delay the start of read from the output buffer and Alarm Interrupt is programmed such that it does not occur while the image header is getting written. This will ensure that the JPEG encoder gets sufficient time to write header information to the output buffer. The JPEG header writes of the new frame will unlock the bus for Cross-bar system bus read.

b) SW Triggered GO bit (VE_MODE[AUTOCLR_GO]=1)

To start write accesses the SW needs to trigger the VE_MODE[AUTOCLR_GO] bit.

Flash memory is not available during the Cross-bar system bus access to the Video Encoder output buffer access is stalled which cause that no software can be executed. In this case VE_MODE[AUTOCLR_GO] bit is not triggered and the lock does not resolve.

e6239: eTimer: When counter is not enabled, capture flags can be set but capture register values are not updated.

Errata type: Errata

Description: If counter is not enabled (ETIMER_CHn_CTRL1[CNTMODE]==000) and a capture event happens (as defined by ETIMER_CHn_CCCTRL[CPTxMODE]), capture flags (ETIMER_CHn_STS[ICF1 or ICF2]) are set. But at the same time the capture registers (ETIMER_CHn_CAPTx) are not updated.

Workaround: There is no workaround for this. The user's software should make sure not to initiate anything using the ETIMER_CHn_STS[ICF1] or ETIMER_CHn_STS[ICF2] flags if the eTimer module is disabled.

e6802: eTimer: Extra input capture events can set unwanted DMA requests

Errata type: Errata

Description: When using the DMA to read the eTimer channel capture registers (ETIMER_CHn_CAPTn) and the DMA has completed its programmed number of transfers an extra input capture event will set the eTimers input capture flag bit in the status register (ETIMER_CHn_STS[ICFn]) and also set the internal DMA request signal. While the input capture flag status bits (ICFn) can be cleared by writing a 1 to their bit positions the DMA request can only be cleared by the DMA done signal. This means that when a new DMA transfer is programmed the eTimer will request a DMA read with possibly unwanted data.

This behavior occurs once the DMA requests are disabled on the side of eDMA (DMA_ERQ[ERQn] = 0), but are still enabled in eTimer (ETIMER_CHn_INTDMA[ICFnDE] = 1), and the active edge is detected

Workaround: In cases where extra eTimer input capture events might occur the following procedure can be used to prevent unwanted DMA read requests:

1. Upon completion of the DMA transfer, disable the DMA requests by clearing the ETIMER_CHn_INTDMA[ICFnDE] bits.

2. If ETIMER_CHn_STS[ICFn] bits are clear then there are no extra input capture events and the eTimer is ready for further operation.

3. If the ICFn bits are set then read the ETIMER_CHn_CAPTn registers until the ETIMER_CHn_CTRL3[CnFCNT] fields are both 0 indicating the capture FIFO's are empty. Then write a 1 to the ICFn bits to clear them. Next, create a dummy DMA read transfer to read the CAPTn registers. The DMA done signal will clear any pending DMA request.

e6583: eTimer: Incorrect updating of the Hold Register

Errata type: Errata

Description: The eTimer's Hold Registers (ETIMER_CHn_HOLD) are incorrectly updated when a Compare and Capture Control Register (ETIMER_CHn_CCCTRL) is read.

Workaround: The eTimer's Hold Registers are supposed to be updated with the Counter Registers (ETIMER_CHn_CNTR) value whenever a Counter Register is read. Recognize that the Hold Registers values will also be updated if a Compare and Capture Control Register is read.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number MPC5604E_1N10D
Revision 10 Nov 2014