Mask Set Errata for Mask 0N72K

This report applies to mask 0N72K for these products:

- MKV40FxxxVLy15
- MKV41FxxxVLy15
- MKV43FxxxVLy15
- MKV44FxxxVLy15
- MKV45FxxxVLy15
- MKV46FxxxVLy15

Table 1. Errata and Information Summary

<table>
<thead>
<tr>
<th>Erratum ID</th>
<th>Erratum Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>e8210</td>
<td>ADC: ADC can't enter lp mode by setting ROSB when JTAG is active.</td>
</tr>
<tr>
<td>e6990</td>
<td>CJTAG: possible incorrect TAP state machine advance during Check Packet</td>
</tr>
<tr>
<td>e6939</td>
<td>Core: Interrupted loads to SP can cause erroneous behavior</td>
</tr>
<tr>
<td>e6940</td>
<td>Core: VDIV or VSQRT instructions might not complete correctly when very short ISRs are used</td>
</tr>
<tr>
<td>e7352</td>
<td>DSPI: reserved bits in slave CTAR are writable</td>
</tr>
<tr>
<td>e8162</td>
<td>I2C: IAAS and IICIF bits in the I2C Status Register are not set properly under certain conditions after low power recovery.</td>
</tr>
<tr>
<td>e7993</td>
<td>MCG: FLL frequency may be incorrect after changing the FLL reference clock</td>
</tr>
<tr>
<td>e7735</td>
<td>MCG: IREFST status bit may set before the IREFS multiplexor switches the FLL reference clock</td>
</tr>
<tr>
<td>e7914</td>
<td>PIT: After enabling the Periodic Interrupt Timer (PIT) clock gate, an attempt to immediately enable the PIT module may not be successful.</td>
</tr>
<tr>
<td>e8209</td>
<td>SOC: High current in VLL3 and VLL2 mode</td>
</tr>
<tr>
<td>e4647</td>
<td>UART: Flow control timing issue can result in loss of characters if FIFO is not enabled</td>
</tr>
<tr>
<td>e2580</td>
<td>UART: Start bit sampling not compliant with LIN 2.1 specification</td>
</tr>
</tbody>
</table>
### Table 2. Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial revision</td>
</tr>
<tr>
<td>16FEB2015</td>
<td>No changes in this revision</td>
</tr>
</tbody>
</table>

**e8210: ADC: ADC can't enter lp mode by setting ROSB when JTAG is active.**

**Description:** ADC can't enter lp mode by setting ROSB when JTAG is connected to debug code

**Workaround:** Recommend to disconnect JTAG, or use SWD.

**e6990: CJTAG: possible incorrect TAP state machine advance during Check Packet**

**Description:** While processing a Check Packet, the IEEE 1149.7 module (CJTAG) internally gates the TCK clock to the CJTAG Test Access Port (TAP) controller in order to hold the TAP controller in the Run-Test-Idle state until the Check Packet completes. A glitch on the internally gated TCK could occur during the transition from the Preamble element to the first Body element of Check Packet processing that would cause the CJTAG TAP controller to change states instead of remaining held in Run-Test-Idle

If the CJTAG TAP controller changes states during the Check Packet due to the clock glitch, the CJTAG will lose synchronization with the external tool, preventing further communication.

**Workaround:** To prevent the possible loss of JTAG synchronization, when processing a Check Packet, provide a logic 0 value on the TMS pin during the Preamble element to avoid a possible glitch on the internally gated TCK clock.

**e6939: Core: Interrupted loads to SP can cause erroneous behavior**

**Description:** ARM Errata 752770: Interrupted loads to SP can cause erroneous behavior

This issue is more prevalent for user code written to manipulate the stack. Most compilers will not be affected by this, but please confirm this with your compiler vendor. MQX™ and FreeRTOS™ are not affected by this issue.

**Affects:** Cortex-M4, Cortex-M4F

**Fault Type:** Programmer Category B

**Fault Status:** Present in: r0p0, r0p1 Open.

If an interrupt occurs during the data-phase of a single word load to the stack-pointer (SP/R13), erroneous behavior can occur. In all cases, returning from the interrupt will result in the load instruction being executed an additional time. For all instructions performing an update to the base register, the base register will be erroneously updated on each execution, resulting in the stack-pointer being loaded from an incorrect memory location.

The affected instructions that can result in the load transaction being repeated are:
1) LDR SP,[Rn],#imm
2) LDR SP,[Rn,#imm]!
3) LDR SP,[Rn,#imm]
4) LDR SP,[Rn]
5) LDR SP,[Rn,Rm]

The affected instructions that can result in the stack-pointer being loaded from an incorrect memory address are:
1) LDR SP,[Rn],#imm
2) LDR SP,[Rn,#imm]!

Conditions:
1) An LDR is executed, with SP/R13 as the destination.
2) The address for the LDR is successfully issued to the memory system.
3) An interrupt is taken before the data has been returned and written to the stack-pointer.

Implications:
Unless the load is being performed to Device or Strongly-Ordered memory, there should be no implications from the repetition of the load. In the unlikely event that the load is being performed to Device or Strongly-Ordered memory, the repeated read can result in the final stack-pointer value being different than had only a single load been performed.

 Interruption of the two write-back forms of the instruction can result in both the base register value and final stack-pointer value being incorrect. This can result in apparent stack corruption and subsequent unintended modification of memory.

Workaround: Most compilers are not affected by this, so a workaround is not required.

However, for hand-written assembly code to manipulate the stack, both issues may be worked around by replacing the direct load to the stack-pointer, with an intermediate load to a general-purpose register followed by a move to the stack-pointer.

If repeated reads are acceptable, then the base-update issue may be worked around by performing the stack pointer load without the base increment followed by a subsequent ADD or SUB instruction to perform the appropriate update to the base register.

e6940: Core: VDIV or VSQRT instructions might not complete correctly when very short ISRs are used

Description: ARM Errata 709718: VDIV or VSQRT instructions might not complete correctly when very short ISRs are used

Affects: Cortex-M4F

Fault Type: Programmer Category B

Fault Status: Present in: r0p0, r0p1 Open.

On Cortex-M4 with FPU, the VDIV and VSQRT instructions take 14 cycles to execute. When an interrupt is taken a VDIV or VSQRT instruction is not terminated, and completes its execution while the interrupt stacking occurs. If lazy context save of floating point state is enabled then the automatic stacking of the floating point context does not occur until a floating point instruction is executed inside the interrupt service routine.
Lazy context save is enabled by default. When it is enabled, the minimum time for the first instruction in the interrupt service routine to start executing is 12 cycles. In certain timing conditions, and if there is only one or two instructions inside the interrupt service routine, then the VDIV or VSQRT instruction might not write its result to the register bank or to the FPSCR.

**Workaround:** A workaround is only required if the floating point unit is present and enabled. A workaround is not required if the memory system inserts one or more wait states to every stack transaction.

There are two workarounds:

1) Disable lazy context save of floating point state by clearing LSPEN to 0 (bit 30 of the FPCCR at address 0xE000EF34).

2) Ensure that every interrupt service routine contains more than 2 instructions in addition to the exception return instruction.

---

**e7352: DSPI: reserved bits in slave CTAR are writable**

**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.

---

**e8162: I2C: IAAS and IICIF bits in the I2C Status Register are not set properly under certain conditions after low power recovery.**

**Description:** The IAAS and IICIF bits in the I2C Status Register indicating an address match and a pending interrupt will not be set if recovering from a low power mode and the Bus Clock is less than ¼ of the Core Clock. Only the TCF flag will be set.

**Workaround:** When recovering from LLSx/VLLSx modes, the Asynchronous Wake-up Interrupt Controller (AWIC) correctly identifies the source of wakeup and can be used to confirm valid I2C addressing. For other low power modes, the TCF flag in conjunction with a software semaphore that establishes recovery from low power modes can be used to determine address match.

---

**e7993: MCG: FLL frequency may be incorrect after changing the FLL reference clock**

**Description:** When the FLL reference clock is switched between the internal reference clock and the external reference clock, the FLL may jump momentarily or lock at a higher than configured frequency. The higher FLL frequency can affect any peripheral using the FLL clock as its input clock. If the FLL is being used as the system clock source, FLL Engaged Internal (FEI) or FLL Engaged External (FEE), the maximum system clock frequency may be exceeded and can cause indeterminate behavior.
Only transitions from FLL External reference (FBE, FEE) to FLL Internal reference (FBI, FEI) modes and vice versa are affected. Transitions to and from BLPI, BLPE, or PLL clock modes (if supported) are not affected because they disable the FLL. Transitions between the external reference modes or between the internal reference modes are not affected because the reference clock is not changed.

**Workaround:** To prevent the occurrence of this jump in frequency either the MCG_C4[DMX32] bit must be inverted or the MCG_C4[DRST DRS] bits must be modified to a different value immediately before the change in reference clock is made and then restored back to their original value after the MCG_S[IREFST] bit reflects the selected reference clock.

If you want to change the MCG_C4[DMX32] or MCG_C4[DRST_DRS] to new values along with the reference clock, the sequence described above must be performed before setting these values to the new value(s).

**e7735:** **MCG: IREFST status bit may set before the IREFS multiplexor switches the FLL reference clock**

**Description:** When transitioning from MCG clock modes FBE or FEE to either FBI or FEI, the MCG_S[IREFST] bit will set to 1 before the IREFS clock multiplexor has actually selected the slow IRC as the reference clock. The delay before the multiplexor actually switches is:

\[2 \text{ cycles of the slow IRC} + 2 \text{ cycles of OSCERCLK}\]

In the majority of cases this has no effect on the operation of the device.

**Workaround:** In the majority of applications no workaround is required. If there is a requirement to know when the IREFS clock multiplexor has actually switched, and OSCERCLK is no longer being used by the FLL, then wait the equivalent time of:

\[2 \text{ cycles of the slow IRC} + 2 \text{ cycles of OSCERCLK}\]

after MCG_S[IREFST] has been set to 1.

**e7914:** **PIT: After enabling the Periodic Interrupt Timer (PIT) clock gate, an attempt to immediately enable the PIT module may not be successful.**

**Description:** If a write to the PIT module enable bit (PIT_MCR[MDIS]) occurs within two bus clock cycles of enabling the PIT clock gate in the SIM_CG register, the write will be ignored and the PIT will fail to enable.

**Workaround:** Insert a read of the PIT_MCR register before writing to the PIT_MCR register. This guarantees a minimum delay of two bus clocks to guarantee the write is not ignored.

**e8209:** **SOC: High current in VLL3 and VLL2 mode**

**Description:** High current in VLL3 and VLL2 mode

**Workaround:** Recommend to use VLLS0, VLLS1 or VLPS mode instead of VLLS2 and VLLS3 mode.
e4647: UART: Flow control timing issue can result in loss of characters if FIFO is not enabled

**Description:** On UARTx modules with FIFO depths greater than 1, when the /RTS flow control signal is used in receiver request-to-send mode, the /RTS signal is negated if the number of characters in the Receive FIFO is equal to or greater than the receive watermark. The /RTS signal will not negate until after the last character (the one that makes the condition for /RTS negation true) is completely received and recognized. This creates a delay between the end of the STOP bit and the negation of the /RTS signal. In some cases this delay can be long enough that a transmitter will start transmission of another character before it has a chance to recognize the negation of the /RTS signal (the /CTS input to the transmitter).

**Workaround:** Always enable the RxFIFO if you are using flow control for UARTx modules with FIFO depths greater than 1. The receive watermark should be set to seven or less. This will ensure that there is space for at least one more character in the FIFO when /RTS negates. So in this case no data would be lost.

Note that only UARTx modules with FIFO depths greater than 1 are affected. The UARTs that do not have the RxFIFO feature are not affected. Check the Reference Manual for your device to determine the FIFO depths that are implemented on the UARTx modules for your device.

e2580: UART: Start bit sampling not compliant with LIN 2.1 specification

**Description:** The LIN 2.1 specification states that start bits should be checked at sample 7, 8, 9, and 10. The UART module checks the start bit at samples 3, 5, and 7 instead.

**Workaround:** Start bits longer than 5/16 of a bit time are guaranteed to be recognized. Start bits shorter than this should not be used with this version of the UART because they might not be recognized.