Mask Set Errata for Mask 0N20V

This report applies to mask 0N20V for these products:

- MWCT1016

Table 1. Errata and information summary

Erratum ID	Erratum title
e6939	Core: Interrupted loads to SP can cause erroneous behavior.
e9004	Core: ITM can deadlock when global timestamping is enabled.
e9005	Core: Storing of immediate overlapping exception return operation might vector to incorrect interrupt.
e6940	Core: VDIV or VSQRT instructions may not complete correctly when very short ISRs are used.
e11450	FLASH: FLASH/EEPROM: Executing quick write functionality on devices using CSEc results in KEY_11 not being usable for all CSEc commands.
e10856	FTM: Safe state is not removed from channel outputs after a fault condition ends when SWOCTRL is being used to control the pin.
e10752	LPI2C: The slave busy flag may incorrectly read as zero when a 10-bit address mode is enabled in the slave mode.
e10792	LPI2C: The slave transmit data flag may be incorrectly read as 1 when TXCFG is 0.
e11097	LPSPI: The command word is not loaded correctly when TXMSK=1.
e11089	LPSPI: In the continuous transfer mode with CPHA =1, the WCF bit is not set for every word.
e11109	MPU: Protection of QSPI memory region requires two Region Descriptors [RGDx].
e10716	RTC: Timer Alarm Flag can assert erroneously.
e10777	SCG: Corrupted status when the system clock is switching.
e11063	SMC: An asynchronous wakeup event during VLPS mode entry may result in possible system hang scenario.
e11114	SMC: invalid data may be fetched while accessing flash in VLP modes.

Table 2. Revision history

Revision	Changes
10/11/2018	Initial revision

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 the user code written to manipulate the stack. Most compilers are not affected by this. However, 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 loaded to the stack-pointer (SP/R13), erroneous behavior can occur. In all cases, returning from the interrupt results in the load instruction being executed for additional time. For all instructions performing an update to the base register, the base register is 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:

- LDR SP,[Rn],#imm
- LDR SP,[Rn,#imm]!
- LDR SP,[Rn,#imm]
- LDR SP,[Rn]
- LDR SP,[Rn,Rm]

The affected instructions that can result in the stack-pointer being loaded from an incorrect memory address are:

- LDR SP,[Rn],#imm 2) LDR SP,[Rn,#imm]!

Conditions:

- An LDR is executed with SP/R13 as the destination.
- The address for the LDR is successfully issued to the memory system.
- An interrupt is taken before the data is returned and written to the stack-pointer.

Implications:

- Unless a load to the device or strongly-ordered memory is performed, there should be no implications from the repetition of the load. In the unlikely event when the load is being performed to the 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.
- An interruption of the two write-back forms of the instruction can result in both the base register value and the 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, 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.

e9004: Core: ITM can deadlock when global timestamping is enabled.

Description: Arm Errata 806422.

The Cortex-M4 processor contains an optional Instrumentation Trace Macrocell (ITM). This can be used to generate trace data under software control and it is also used with the Data Watchpoint and Trace (DWT) module which generates an event-driven trace. The processor supports global timestamping. This allows the counting values from a system-wide counter to be included in the trace stream.

When connected directly to a CoreSight funnel (or other component that holds ATREADY low in the idle state), the ITM stops presenting the trace data to the ATB bus after generating a timestamp packet. In this condition, the ITM_TCR.BUSY register indicates BUSY.

When this condition occurs, a reset of the Cortex-M4 core is necessary before new trace data can be generated by the ITM.

The timestamp packets that require a 5-byte GTS1 packet or a GTS2 packet do not trigger this erratum. This generally only applies to the first timestamp generated.

Devices that use the Cortex-M optimized TPIU (CoreSight ID register values 0x923 and 0x9A1) are not affected by this erratum.

Workaround: There is no software workaround for this erratum. If the device used is susceptible to this erratum, do not enable global timestamping.

e9005: Core: Storing of immediate overlapping exception return operation might vector to incorrect interrupt.

Description: Arm Errata 838869: Storing of immediate overlapping exception return operation might vector to incorrect interrupt.

Affects: Cortex-M4, Cortex-M4F.

Fault type: Programmer category B rare fault status: present in: r0p0, r0p1 open.

Cortex-M4 includes a write buffer that permits the execution to continue while a store is waiting on the bus. Under specific timing conditions (during an exception return while this buffer is still in use by a store instruction), a late change in selection of the next interrupt to be taken may result in a mismatch between the interrupt acknowledged by the interrupt controller and the vector fetched by the processor.

Configurations affected: This erratum only affects systems where the writeable memory locations can exhibit more than one wait state.

Workaround: For the software not using the memory protection unit, this erratum can be worked around by setting DISDEFWBUF in the auxiliary control register.

In all other cases, the erratum can be avoided by ensuring a DSB occurs between the store and the BX instruction. For exception handlers written in C, this can be achieved by inserting the appropriate set of intrinsics or inline assembly just before the end of the interrupt function; for example:

- ARMCC:

```
...
__schedule_barrier();
__asm{DSB};
__schedule_barrier();
}
GCC:
...
__asm volatile (“dsb 0xf” ::: “memory”);
}
```

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

Description: Arm Errata 709718: VDIV or VSQRT instructions may 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 an 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 a lazy context save of the floating point state is enabled, the automatic stacking of the floating point context does not occur until a floating point instruction is executed inside the interrupt service routine.

The 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, the VDIV or VSQRT instructions might not write their results to the register bank or 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:

- Disable the lazy context save of the floating point state by clearing LSPEN to 0 (bit 30 of the FPCCR at address 0xE000EF34).
- Ensure that every interrupt service routine contains more than two instructions in addition to the exception return instruction.

e11450: FLASH: FLASH/EEPROM: Executing quick write functionality on devices using CSEc results in KEY_11 not being usable for all CSEc commands.

Description: Executing quick write functionality on devices using CSEc corrupts the attributes associated with KEY_11, impacting the possibilities of using KEY_11 as a part of any CSEc command. KEY_11 integrity and confidentiality are not compromised.

Workaround: Do not use quick write with a CSEc feature or:

- The EEPROM must be placed into the normal write mode (using command SETRAM with option 0x00) before any CSEc functionality using KEY_11.
- Do not use KEY_11 (use other keys instead of KEY_11 if available).

e10856: FTM: Safe state is not removed from channel outputs after a fault condition ends when SWOCTRL is being used to control the pin.

Description: If the FTM channel output is controlled using the software output control register (FTM_SWOCTRL) and the fault detection is also enabled for the channel, the output is forced to its safe value when a fault is detected. However, when the fault condition is cleared, the channel output stays in the safe state instead of reverting to the value programmed by the FTM_SWOCTRL register.

Workaround: If the fault control is enabled while the software output control register is also being used (FTM_SWOCTRL), the FTM must be configured as follows:

- FTM_MODE[FAULTM] configured for manual fault clearing (0b10).
- For devices that include the FTM_CONF[NUMTOF] field, it must be cleared to 0b00000 (TOF set for each counter overflow). For FTM versions that don't include the FTM_CONF[NUMTOF] field, this does not apply.
- The below procedure must be used in the TOF interrupt handler when a fault is detected to ensure that the outputs return to the value configured by FTM_SWOCTRL.

1. Check the value of FTM_FMS[FAULTF]:
    - If FTM_FMS[FAULTF] = 1 (a fault occurred or is occurring), set a variable to indicate that a fault was detected and continue to step 2.
    - If FTM_FMS[FAULTF] = 0 but the fault variable is set (a fault is not active, but was previously detected), continue to step 6.
2. Write the FTM_OUTMASK register to set the bit or bits corresponding to any channels that are controlled by FTM_SWOCTRL to temporarily deactivate the channel output.
3. Clear the fault conditions by reading the FTM_FMS register and then writing FTM_FMS with all zeroes.
4. Clear the FTM_SC[TOF] bit by reading the FTM_SC register and then writing 0 to FTM_SC[TOF].
5. Exit the interrupt handler to skip the following steps (they execute the next time the TOFhandler is called).
6. Clear FTM_SWOCTRL by writing all zeroes to it.
7. Write FTM_SWOCTRL with the desired value again.
8. Clear the FTM_OUTMASK bits that were set in step 2.
9. Clear the fault variable that was set in step 1 when the fault condition was originally detected.
10. Clear the FTM_SC[TOF] bit by reading the FTM_SC register and write 0 to FTM_SC[TOF].

e10752: LPI2C: The slave busy flag may incorrectly read as zero when a 10-bit address mode is enabled in the slave mode.

Description: When operating in the slave mode with a 10-bit addressing enabled and detecting an address match on the first address byte, the slave busy flag (SSR[SBF]) may incorrectly read as 0.

Workaround: When using the LPI2C in the slave mode when a 10-bit addressing is enabled and when the SSR[SBF] bit is read as a 1, the flag is set. If it is read as 0, it must be read for a second time and this second read is the correct state of the bit.

e10792: LPI2C: The slave transmit data flag may be incorrectly read as 1 when TXCFG is 0.

Description: When SCFGR1[TXCFG] = 0, the slave transmit data ready flag can incorrectly assert for one cycle.

Workaround: Set SCFGR1[TXCFG] = 1.

e11097: LPSPI: The command word is not loaded correctly when TXMSK=1.

Description: When the transmit command register is written with TCR[TXMSK]=1 and the next write to the TX FIFO is another command, the first command may not load correctly.

Workaround: When writing the transmit command register with TCR[TXMSK]=1, wait for the TX FIFO to empty (FSR[TXCOUNT] = 0) before writing another command to the transmit command register.

e11089: LPSPI: In the continuous transfer mode with CPHA =1, the WCF bit is not set for every word.

Description: When the transmit command register is written with TCR[CONT]=1 and TCR[CPHA]=1, the SR[WCF] bit flag is not set after the data is transferred. Therefore, polling for the SR[WCF] flag to identify whether the sent data can cause the MCU to get stuck.

Workaround: When using the continuous transfer mode TCR[CONT]=1 and TCR[CPHA]=1, do not use the SR[WCF] flag to determine whether the data were sent and fill up the transmit FIFO with the following data without waiting for the SR[WCF] flag to be set.

e11109: MPU: Protection of QSPI memory region requires two Region Descriptors [RGDx].

Description: The MPU requires a special programming sequence to protect the QSPI space because it is not able to see the two MSB bits of the QSPI address on slave port 4. This programming sequence requires two region descriptors [RGDx].

Workaround: To properly provide protection to the QSPI space, it is necessary to use two RGDs. One covers region 0x280x_xxxx and the other one covers region 0x680x_xxxx. When any master tries to access region 0x680x_xxxx without permissions, an error is captured in both registers (EDR3 and EDR4). Moreover, the address of the failed access is captured in the EAR3 and EAR4 registers. However, EAR3 captures address 0x680x_xxxx, which is the address that belongs to the QSPI space while EAR4 captures the 0x280x_xxxx address.

e10716: RTC: Timer Alarm Flag can assert erroneously.

Description: Writing to the time alarm register (RTC_TAR) while the RTC seconds register is incrementing can assert the time alarm flag (RTC_SR[TAF]) bit in the RTC status register.

Workaround: Write the time alarm register (RTC_TAR) when the RTC seconds register is not incrementing. This happens when the time counter enable (RTC_SR[TCE]) bit in the RTC status register is clear or within the RTC_SR[TAF] interrupt routine.

Alternatively, if RTC_SR[TAF] is asserted after a write to RTC_TAR, write the RTC_TAR again.

e10777: SCG: Corrupted status when the system clock is switching.

Description: SCG_RCCR[SCS] and SCG_HCCR[SCS] may have corrupted status in the interval when the system clock is switching.

Workaround: The SCS field must be read twice by the software to ensure that the system clock switch completed.

e11063: SMC: An asynchronous wakeup event during VLPS mode entry may result in possible system hang scenario.

Description: When the bus clock is also the system clock and an asynchronous wakeup occurs during a mode transition from RUN to VLPS or VLPR to VLPS, the MCU may hang in an undetermined state, which can only be recovered by a power-on reset event or a watchdog reset.

Workaround: Before executing the transition to VLPS, ensure that the PREDIV_SYS_CLK frequency/BUS_CLK frequency configuration for the RUN/VLPR mode is greater than or equal to 2; for example, assuming PREDIV_SYS_CLK of 8 MHz and SCG_RCCR[DIVCORE] = 0b0001 (a divider of 2) and SCG_RCCR[DIVBUS] = 0b0001 (a divider of 1), (PREDIV_SYS_CLK = 8 MHz)/(BUS_CLK = 4 MHz), a ratio of 1:2.

e11114: SMC: invalid data may be fetched while accessing flash in VLP modes.

Description: The VLPR and VLPS low-power modes are documented to work with the system clock and the core clock at 4 MHz, the bus clock at 4 MHz, and the DMA enabled from (or to) the flash memory. However, any simultaneous access from any master (core or DMA) to the Dflash and Pflash may get invalid data while being in the VLP modes with the system clock, core clock, and bus clock above 1 MHz.

Workaround: There are two workarounds:

- Restrict the software to use either only Pflash or Dflash at a time in the VLP modes for all masters (CPU and DMA). When switching from the Pflash-only access to the Dflash-only access, let the current DMA transactions that access the flash complete, jump to the SRAM location, and wait for 40 cycles for the ongoing accesses to complete on the current flash before accessing the Dflash.
- When switching from the Dflash-only accesses to the Pflash-only accesses, let the current DMA transactions that access the Dflash complete and wait for 40 cycles for the accesses to complete on the Dflash before accessing the Pflash.

If both the Pflash and Dflash must be accessed simultaneously, the VLP modes must be run with the system clock, core clock, and bus clock at 1 MHz.

How to Reach Us:

Home Page:
www.nxp.com

Web Support:
www.nxp.com/support

Document Number: MWCT1016_0N20V
Rev. 10/Nov/2018