Mask Set Errata for Mask 1N41K

This document contains errata information for Kinetis Mask Set 1N41K but excludes any information on selected security-related modules.

A nondisclosure agreement (NDA) is required for any security-related module information.

For more information on obtaining an NDA, please contact your local Freescale sales representative.

Mask Set Errata for Mask 1N41K

This report applies to mask 1N41K for these products:

- KINETIS

Errata ID	Errata Title
6990	CJTAG: possible incorrect TAP state machine advance during Check Packet
6939	Core: Interrupted loads to SP can cause erroneous behavior
6940	Core: VDIV or VSQRT instructions might not complete correctly when very short ISRs are used
8010	LLWU: CMP flag in LLWU_Fx register cleared by multiple CMP out toggles when exiting LLSx or VLLSx modes.
7993	MCG: FLL frequency may be incorrect after changing the FLL reference clock
7735	MCG: IREFST status bit may set before the IREFS multiplexor switches the FLL reference clock
5130	SAI: Under certain conditions, the CPU cannot reenter STOP mode via an asynchronous interrupt wakeup event
3981	SDHC: ADMA fails when data length in the last descriptor is less or equal to 4 bytes
3982	SDHC: ADMA transfer error when the block size is not a multiple of four
4627	SDHC: Erroneous CMD CRC error and CMD Index error may occur on sending new CMD during data transfer
3983	SDHC: Problem when ADMA2 last descriptor is LINK or NOP
4218	SIM/FLEXBUS: SIM_SCGC7[FLEXBUS] bit should be cleared when the FlexBus is not being used.
4647	UART: Flow control timing issue can result in loss of characters if FIFO is not enabled
8169	UART: ISO-7816 T=0 mode loss of characters when UART is switched from transmit to receive mode
6933	eDMA: Possible misbehavior of a preempted channel when using continuous link mode

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

Errata type: Errata

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

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

Affects: Cortex-M4, Cortex-M4F

Fault Type: Programmer Category B

Fault Status: Present in: r0p0, r0p1 Open.

Description

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: 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

Errata type: Errata

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.

e8010: LLWU: CMP flag in LLWU_Fx register cleared by multiple CMP out toggles when exiting LLSx or VLLSx modes.

Errata type: Errata

Description: The comparator's corresponding wakeup flag in the LLWU_Fx register is cleared prematurely if:

1. The CMP output is toggled more than one time during the LLSx wakeup sequence and the comparator's corresponding flag in the LLWU_Fx register is cleared.

Or

2. The CMP output is toggled more than one time during the VLLSx wakeup sequence, PMC_REGSC[ACKISO] is cleared, and the comparator's corresponding flag in the LLWU_Fx register is cleared.

Workaround: When MCU is waking up from LLS, code can implement a software flag to retain the wakeup source, if required by software.

When MCU is waking up from VLLSx, code can implement a software flag prior to clearing PMC_REGSC[ACKISO] to retain the wakeup source, if required by software.

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

Errata type: Errata

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

Errata type: Errata

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 cycles of the slow IRC + 2 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 cycles of the slow IRC + 2 cycles of OSCERCLK

after MCG_S[IREFST] has been set to 1.

e5130: SAI: Under certain conditions, the CPU cannot reenter STOP mode via an asynchronous interrupt wakeup event

Errata type: Errata

Description: If the SAI generates an asynchronous interrupt to wake the core and it attempts to reenter STOP mode, then under certain conditions the STOP mode entry is blocked and the asynchronous interrupt will remain set.

This issue applies to interrupt wakeups due to the FIFO request flags or FIFO warning flags and then only if the time between the STOP mode exit and subsequent STOP mode reentry is less than 3 asynchronous bit clock cycles.

Workaround: Ensure that at least 3 bit clock cycles elapse following an asynchronous interrupt wakeup event, before STOP mode is reentered.

e3981: SDHC: ADMA fails when data length in the last descriptor is less or equal to 4 bytes

Errata type: Errata

Description: A possible data corruption or incorrect bus transactions on the internal AHB bus, causing possible system corruption or a stall, can occur under the combination of the following conditions:

1. ADMA2 or ADMA1 type descriptor

2. TRANS descriptor with END flag

3. Data length is less than or equal to 4 bytes (the length field of the corresponding descriptor is set to 1, 2, 3, or 4) and the ADMA transfers one 32-bit word on the bus

4. Block Count Enable mode

Workaround: The software should avoid setting ADMA type last descriptor (TRANS descriptor with END flag) to data length less than or equal to 4 bytes. In ADMA1 mode, if needed, a last NOP descriptor can be appended to the descriptors list. In ADMA2 mode this workaround is not feasible due to ERR003983.

e3982: SDHC: ADMA transfer error when the block size is not a multiple of four

Errata type: Errata

Description: Issue in eSDHC ADMA mode operation. The eSDHC read transfer is not completed when block size is not a multiple of 4 in transfer mode ADMA1 or ADMA2. The eSDHC DMA controller is stuck waiting for the IRQSTAT[TC] bit in the interrupt status register.

The following examples trigger this issue:

1. Working with an SD card while setting ADMA1 mode in the eSDHC

2. Performing partial block read

3. Writing one block of length 0x200

4. Reading two blocks of length 0x22 each. Reading from the address where the write operation is performed. Start address is 0x512 aligned. Watermark is set as one word during read. This read is performed using only one ADMA1 descriptor in which the total size of the transfer is programmed as 0x44 (2 blocks of 0x22).

Workaround: When the ADMA1 or ADMA2 mode is used and the block size is not a multiple of 4, the block size should be rounded to the next multiple of 4 bytes via software. In case of write, the software should add the corresponding number of bytes at each block end, before the write is initialized. In case of read, the software should remove the dummy bytes after the read is completed.

For example, if the original block length is 22 bytes, and there are several blocks to transfer, the software should set the block size to 24. The following data is written/stored in the external memory:

4 Bytes valid data

4 Bytes valid data

4 Bytes valid data

4 Bytes valid data

4 Bytes valid data

2 Bytes valid data + 2 Byte dummy data

4 Bytes valid data

4 Bytes valid data

4 Bytes valid data

4 Bytes valid data

4 Bytes valid data

2 Bytes valid data + 2 Byte dummy data

In this example, 48 (24 x 2) bytes are transferred instead of 44 bytes. The software should remove the dummy data.

e4627: SDHC: Erroneous CMD CRC error and CMD Index error may occur on sending new CMD during data transfer

Errata type: Errata

Description: When sending new, non data CMD during data transfer between the eSDHC and EMMC card, the module may return an erroneous CMD CRC error and CMD Index error. This occurs when the CMD response has arrived at the moment the FIFO clock is stopped. The following bits after the start bit of the response are wrongly interpreted as index, generating the CRC and Index errors.

The data transfer itself is not impacted.

The rate of occurrence of the issue is very small, as there is a need for the following combination of conditions to occur at the same cycle:

- The FIFO clock is stopped due to FIFO full or FIFO empty
- The CMD response start bit is received

Workaround: The recommendation is to not set FIFO watermark level to a too small value in order to reduce frequency of clock pauses.

The problem is identified by receiving the CMD CRC error and CMD Index error. Once this issue occurs, one can send the same CMD again until operation is successful.

e3983: SDHC: Problem when ADMA2 last descriptor is LINK or NOP

Errata type: Errata

Description: ADMA2 mode in the eSDHC is used for transfers to/from the SD card. There are three types of ADMA2 descriptors: TRANS, LINK or NOP. The eSDHC has a problem when the last descriptor (which has the End bit '1') is a LINK descriptor or a NOP descriptor.

In this case, the eSDHC completes the transfers associated with this descriptor set, whereas it does not even start the transfers associated with the new data command. For example, if a WRITE transfer operation is performed on the card using ADMA2, and the last descriptor of the WRITE descriptor set is a LINK descriptor, then the WRITE is successfully finished. Now, if a READ transfer is programmed from the SD card using ADMA2, then this transfer does not go through.

Workaround: Software workaround is to always program TRANS descriptor as the last descriptor.

e4218: SIM/FLEXBUS: SIM_SCGC7[FLEXBUS] bit should be cleared when the FlexBus is not being used.

Errata type: Errata

Description: The SIM_SCGC7[FLEXBUS] bit is set by default. This means that the FlexBus will be enabled and come up in global chip select mode.

With some code sequence and register value combinations the core could attempt to prefetch from the FlexBus even though it might not actually use the value it prefetched. In the case where the FlexBus is unconfigured, this can result in a hung bus cycle on the FlexBus.

Workaround: If the FlexBus is not being used, disabled the clock to the FlexBus during chip initialization by clearing the SIM_SCGC7[FLEXBUS] bit.

If the FlexBus will be used, then enable at least one chip select as early in the chip initialization process as possible.

e4647: UART: Flow control timing issue can result in loss of characters if FIFO is not enabled

Errata type: Errata

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.

e8169: UART: ISO-7816 T=0 mode loss of characters when UART is switched from transmit to receive mode

Errata type: Errata

Description: When operating in ISO-7816 T=0 mode, S1[TC] sets to indicate end of transmission after 12 ETUs and software then switches the UART to receive mode by setting C2[RE]. Delay between the transmit and receive transition may cause the start bit of an incoming character to be missed resulting in a missed character.

Workaround: Software should switch the UART to receive mode by setting C2[RE] within 0.3 ETUs of S1[TC] being set. No workaround is required for EMV card applications because the maximum turnaround time for EMV-compliant cards is 15 ETUs, per the EMV L1 test specification (1CF. 004.00).

e6933: eDMA: Possible misbehavior of a preempted channel when using continuous link mode

Errata type: Errata

Description: When using continuous link mode (DMA_CR[CLM] = 1) with a high priority channel linking to itself, if the high priority channel preempts a lower priority channel on the cycle before its last read/write sequence, the counters for the preempted channel (the lower priority channel) are corrupted. When the preempted channel is restored, it runs past its "done" point instead of performing a single read/write sequence and retiring.

The preempting channel (the higher priority channel) will execute as expected.

Workaround: Disable continuous link mode (DMA_CR[CLM]=0) if a high priority channel is using minor loop channel linking to itself and preemption is enabled. The second activation of the preempting channel will experience the normal startup latency (one read/write sequence + startup) instead of the shortened latency (startup only) provided by continuous link mode.

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.