



ABSTRACT

This document describes the known exceptions to the functional specifications (advisories).

Table of Contents

1 Functional Advisories	1
2 Preprogrammed Software Advisories	1
3 Debug Only Advisories	1
4 Fixed by Compiler Advisories	2
5 Device Nomenclature	2
5.1 Device Symbolization and Revision Identification.....	2
6 Advisory Descriptions	4
7 Trademarks	9
8 Revision History	9

1 Functional Advisories

Advisories that affect the device operation, function, or parametrics.

✓ The check mark indicates that the issue is present in the specified revision.

Errata Number	Rev A
AES_ERR_01	✓
CPU_ERR_05	✓
GPIO_ERR_05	✓
GSC_ERR_01 GSC Module	✓
IOMUX_ERR_01	✓
KEYSTORE_ERR_01	✓
PMCU_ERR_15	✓
SYSCTL_ERR_01	✓
SYSPLL_ERR_01	✓
TIMER_ERR_04	✓
TIMER_ERR_06	✓
TIMER_ERR_07	✓
TIMER_ERR_08	✓

2 Preprogrammed Software Advisories

Advisories that affect factory-programmed software.

✓ The check mark indicates that the issue is present in the specified revision.

3 Debug Only Advisories

Advisories that affect only debug operation.

✓ The check mark indicates that the issue is present in the specified revision.

4 Fixed by Compiler Advisories

Advisories that are resolved by compiler workaround. Refer to each advisory for the IDE and compiler versions with a workaround.

✓ The check mark indicates that the issue is present in the specified revision.

5 Device Nomenclature

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all MSP MCU devices. Each MSP MCU commercial family member has one of two prefixes: MSP or XMS. These prefixes represent evolutionary stages of product development from engineering prototypes (XMS) through fully qualified production devices (MSP).

XMS – Experimental device that is not necessarily representative of the final device's electrical specifications

MSP – Fully qualified production device

Support tool naming prefixes:

X: Development-support product that has not yet completed Texas Instruments internal qualification testing.

null: Fully-qualified development-support product.

XMS devices and X development-support tools are shipped against the following disclaimer:

"Developmental product is intended for internal evaluation purposes."

MSP devices have been characterized fully, and the quality and reliability of the device have been demonstrated fully. TI's standard warranty applies.

Predictions show that prototype devices (XMS) have a greater failure rate than the standard production devices. TI recommends that these devices not be used in any production system because their expected end-use failure rate still is undefined. Only qualified production devices are to be used.

TI device nomenclature also includes a suffix with the device family name. This suffix indicates the temperature range, package type, and distribution format.

5.1 Device Symbolization and Revision Identification

The package diagrams below indicate the package symbolization scheme, and [Table 5-1](#) defines the device revision to version ID mapping.

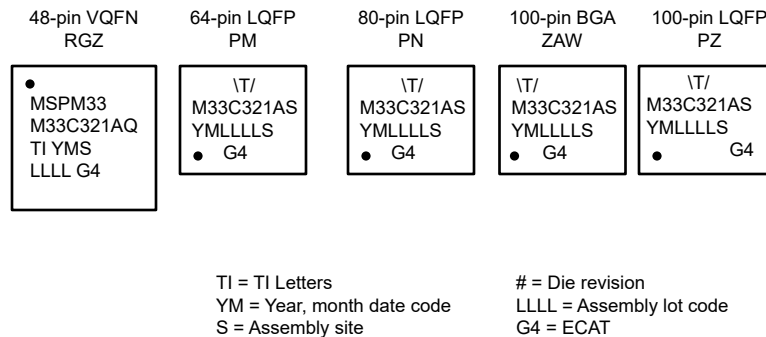


Figure 5-1. Package Symbolization

Table 5-1. Die Revisions

Revision Letter	Version (in the device factory constants memory)
A	0

The revision letter indicates the product hardware revision. Advisories in this document are marked as applicable or not applicable for a given device based on the revision letter. This letter maps to an integer stored in the

memory of the device, which can be used to look up the revision using application software or a connected debug probe.

6 Advisory Descriptions

AES_ERR_01

AES Module

Category

Functional

Function

AES Saved Context Ready interrupt is not generating as expected

Description

Saved Context Ready interrupt is not getting generated. The interrupt is generated if an access (read or write) is made to any AES register.

Workaround

Use polling based mechanism to check the status bit for Saved Context Ready in CTRL register instead of interrupt.

CPU_ERR_05

CPU Module

Category

Functional

Function

MTB_BASE register value gets read improperly

Description

When Reading the MTB_BASE register the MTB_SRAM base location will be right shifted by a bit. For example, if the MTB_SRAM shows 0x80806000 the actual MTB_SRAM base location is 0x40403000.

Workaround

When reading the MTB_BASE register to find the MTB_SRAM base location right shift the read value by one bit. Please use the code below to work around the issue: `uint32_t mtb_base = (*(uint32_t*)(0xE004300C) >> 1);`

GPIO_ERR_05

GPIO Module

Category

Functional

Function

Writing to GPIO DOUTTGL registers might get missed when a DMA transfer is ongoing

Description

The GPIO DMAMASK register information is mistakenly applied to a CPU write to the DOUTTGL register when a concurrent DMA transfer is in progress.

Workaround

In the application code, ensure that the GPIO DMAMASK bit is set to 1 for the corresponding bit in the DOUTTGL register, before a CPU write access to the DOUTTGL register is issued. If no DMA transfer to any of the GPIO registers is required, the GPIO DMAMASK can be configured as 0xFFFFFFFF during the IO initialization step. This will solve the conflict of this errata. If the application also requires DMA write transfers to the GPIO registers, it is recommended that the application not use both DMA and CPU to write to the DOUTTGL register of the same GPIO module in the device. If the device has multiple GPIO modules, the DMA and the CPU can simultaneously write to the DOUTTGL

GPIO_ERR_05

(continued)

GPIO Module

register of different GPIO modules (while still requiring that the GPIO DMAMASK be configured for the GPIO module the CPU is writing to).

GSC_ERR_01

GSC Module

Category

Functional

Function

Certain timers have the same security attribute

Description

In the PPC_SECATTRIB_TIMER register the SEC_TIMA0_0 and SEC_TIMA0_1 registers are tied together such that the security attribute for both TIMERA0_0 and TIMERA0_1 must be the same. TIMERG4_2 and TIMERG4_1 also have the same configuration but only when the DMA is attempting to access the timers.

Workaround

Try to keep both TIMERA0_0 and TIMERA0_1 as the same security level. If that's not possible use a different pair of timer instances. For TIMERG4_2 and TIMERG4_1 keep them at the same security level especially if using a DMA to access the TIMERG4_2 or TIMERG4_1.

IOMUX_ERR_01

IOMUX Module

Category

Functional

Function

TDO pin drives a low signal after reset.

Description

The TDO pin out of reset is driven as a low signal. For other pins they are in a Hi-Z state out of reset.

Workaround

If you need this pin to be a Hi-Z state set both the PC and PF bits to zero. Otherwise connect the peripheral you plan to use on this IO.

KEYSTORE_ERR_01

KEYSTORE Module

Category

Functional

Function

STATUS.STAT value can be 0 or 1 without key access

Description

STATUS.STAT has a reset value of 1 and turns to 0 under these conditions: 1. After reset, debugger access via the register window returns 0x00. 2. After reset, the first CPU read returns 0x01, while subsequent CPU reads return 0x00. 3) After reset, first reading any other KEYSTORE register and then reading STATUS.STAT return 0x00.

KEYSTORE_ERR_
01 (continued) *KEYSTORE Module*

Workaround

STATUS.STAT=0x0 means "No Error" . For checking if a slot is valid or not (Whether key is present), check STATUS.VALID.

PMCU_ERR_15
PMCU Module

Category

Functional

Function

GENCLKEN Clock related configurations before SYSPLL is enabled are ignored

Description

When configuring the SYSPLL, configure the GENCLKEN register after setting the SYSPLLEN bit in the HSCLKEN register. If the EXTDIVCAN, MCLKEXTDIVEN and I2SPLLCLKDIVEN, I2SPLLCLKDIVCFG bits in the GENCLKEN register are programmed before the SYSPLLEN bit is set, the configuration in the GENCLKEN register is not properly configured and the EXTDIVCAN and I2SPLLCLKDIVCFG configurations are ignored.

Workaround

Configure the EXTDIVCAN, MCLKEXTDIVEN and I2SPLLCLKDIVEN, I2SPLLCLKDIVCFG bits in the GENCLKEN registers, after the SYSPLLEN bit is set.

SYSCTL_ERR_01 *SYSCTL Module*

Category

Functional

Function

SW-POR functionality is combined with HW-POR

Description

When a user writes to the LFSSRST register with the correct key to generate a software-triggered POR, the RSTCAUSE register will display 0x2 (indicating an NRST-triggered POR) instead of the expected 0x3 (Software-Triggered POR). This occurs because the SW-POR functionality is combined with the HW-POR path.

Workaround

No

SYSPLL_ERR_01 *SYSPLL Module*

Category

Functional

Function

SYSPLL Frequency may not lock to correct frequency when enabled.

Description

When setting the SYSPLLEN bit to 1 in SYSCTL HSCLKEN register, the SYSPLL will run the phase locked loop search. The search can potentially fail where the frequency will not be set to the correct value, instead the resultant frequency will be drastically different than the configured frequency.

SYSPLL_ERR_01

(continued)

SYSPLL Module

Workaround

Frequency Verification Process

Monitor the SYSPLL frequency output using the Frequency Clock Counter (FCC) whenever the SYSPLLEN bit is set to 1. Once the correct frequency is established, it will remain stable until the SYSPLL is disabled and re-enabled (SYSPLLEN bit toggled from 0 to 1). If an incorrect frequency is detected, disable and re-enable the SYSPLL to perform another verification.

Workaround 1: FCC Count Check

Use LFCLK as the FCC trigger source to count the SYSPLL output clock frequency. Execute the FCC and verify the measured value against the configured SYSPLL frequency using LFCLK as reference.

Example calculation:

- SYSPLLCLK0 = 80MHz ; LFCLK = 32.768kHz
- Measured FCC Count = $80,000,000/32,768 = 2,441$

FCC Count Tolerance:

The real FCC count will vary depending on the combined clock accuracies (SYSPLLCLK0 and LFCLK). Recommend to add +/- 5~10% to allowed FCC check range.

- FCC count upper limit = $2,441 * 1.05 = 2,563$
- FCC count lower limit = $2,441 * 0.95 = 2,318$

Timing considerations:

- Clock synchronization time: 5-6 LFCLK cycles
- FCC trigger time: 1-32 LFCLK cycles (user-configurable)

Register Configuration:

- FCC Settings: SYSCTL.GENCLKCFG.FCCTRIGSRC = 1;
- SYSCTL.GENCLKCFG.FCCLVLRIG = 0;
- SYSCTL.GENCLKCFG.FCCTRIGCNT = 0;
- SYSCTL.GENCLKCFG.FCCSELCLK = 4;
- Start FCC: SYSCTL.FCCCMD = 0x0E000001U
- Check FCC Done Status: SYSCTL.CLKSTATUS.FCCDONE
- Read FCC Count: SYSCTL.FCC

Timeout Protection:

Implement a software-based timeout during FCC Done status monitoring to prevent longer wait time under the condition the unlocked SYSPLL frequency is less than FCC trigger clock frequency (LFCLK).

```
fccTimeOutCounter = 0;
while (DL_SYCTL_isFCCDone() == 0) {
    delay_cycles(977); /* 1x LFCLK cycle = 32MHz/32.768kHz */
    fccTimeOutCounter++;
    if(fccTimeOutCounter > 65) break;
    /* Timeout set to approximately 2ms (user-customizable) */
}
```

FCC Check Restart:

If the FCC measurement falls outside the expected range, disable and re-enable the SYSPLL (set SYSPLLEN to 0, then 1) and repeat the FCC verification.

Workaround 2: FCC Ratio Check

Use LFCLK as the FCC trigger source to count both SYSPLL output and input clock frequency. Execute the FCC and verify the measured ratio of the FCC check value between output and input clock to the expected ratio.

Example calculation:

- SYSPLL = 80MHz ; HFCLK = 40MHz ; LFCLK = 32.768kHz

SYSPLL_ERR_01

(continued)

SYSPLL Module

- Expect clock ratio = 80MHz/40MHz = 2.0000
- Measured FCC count (SYSPLL) = 80,000,000/32,768 = 2,441
- Measured FCC count (HFCLK) = 40,000,000/32,768 = 1,220
- Measured clock ratio = 2,441/1,220 = 2.0008

FCC Ratio Tolerance:

The FCC ratio method eliminates combined clock accuracy errors and depends only on FCC uncertainty (2 counted clock cycles) and calculation rounding error. This allows for much tighter tolerance ranges compared to FCC count check method, for example +/- 0.3%.

Timing considerations:

- Clock synchronization time: 5-6 LFCLK cycles
- FCC trigger time: 1-32 LFCLK cycles (user-configurable)
- Total time per complete FCC ratio check: 2*(sync time + trigger time)

FCC Ratio Check Flow:

1. Configure FCC for SYSPLL output clock (SYSPLL0 or SYSPLL2X)
2. Start FCC and wait for FCC done (Add Timeout Protection, refer to FCC Count Check)
3. Read FCC check count back
4. Configure FCC for SYSPLL input clock (SYSOSC or HFCLK)
5. Start FCC and wait for FCC done (Add Timeout Protection, refer to FCC Count Check)
6. Read FCC check count back
7. Calculate the FCC check ratio and compared to the expected ratio range
8. If the FCC ratio falls outside the expected range, disable and re-enable the SYSPLL (set SYSPLLEN to 0, then 1) and repeat the FCC ratio verification.

TIMER_ERR_04**TIMER Module**

Category

Functional

Function

TIMER re-enable may be missed if done close to zero event

Description

When using a TIMER in one shot mode, TIMER re-enable may be missed if done close to zero event. The HW update to the timer enable bit will take a single functional clock cycle. For example, if the timer's clock source is 32.768kHz and clock divider of 3, then it will take ~100us to have the enable bit set to 0 properly.

Workaround

Wait 1 functional clock cycle before re-enabling the timer OR the timer can be disabled first before re-enabling.

Disable the counter with CTRCTL.EN = 0, then re-enable with CTRCTL.EN = 1

TIMER_ERR_06**TIMER Module**

Category

Functional

Function

Writing 0 to CLKEN bit does not disable counter

TIMER_ERR_06

(continued)

TIMER Module

Description

Writing 0 to the Counter Clock Control Register(CCLKCTL) Clock Enable bit(CLKEN) does not stop the timer.

Workaround

Stop the timer by writing 0 to the Counter Control(CTRCTL) Enable(EN) bit.

TIMER_ERR_07

Initial repeat counter has 1 less period than next repeats Module

Category

Functional

Function

TIMER

Description

When using the timer repeat counter mode, the first repeat will have 1 less count than the subsequent repeats because the following repeat counters will include the transition between 0 and the load value. For example if the TIMx.RCLD = 0x3 then 3 observable zero events would appear on the first repeat counter and 4 observable zero events would appear on the following repeat counter sequences.

Workaround

Set the initial RCLD value to 1 more than the expected RCLD, then in the ISR for the Repeat Counter Zero Event (REPC), set the RCLD to the intended RCLD value. For example, if intending to have 4 repeats, set the initial RCLD value to RCLD = 0x5, then in the timer ISR for the REPC interrupt, set RCLD = 0x4. Now all timer repeats will have the same number of zero/load events.

TIMER_ERR_08

TIMx Module

Category

Functional

Function

Sequential event generation for advanced timers GEN_EVENT1 output is ignored

Description

In the advanced timer's (TIMA) the GEN_EVENT1 only works for one event and will not properly propagate through the event handler after the first event is triggered.

Workaround

Do not use GEN_EVENT1 for repeated event outputs on advanced timers. Either use GEN_EVENT0 or use a general-purpose timer (TIMG) instance instead.

7 Trademarks

All trademarks are the property of their respective owners.

8 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from December 31, 2025 to April 30, 2026 (from Revision * (December 2025) to Revision A (April 2026))
Page

• CPU_ERR_05 Category was updated.....	4
• CPU_ERR_05 Module was updated.....	4
• CPU_ERR_05 Description was updated.....	4
• CPU_ERR_05 Function was updated.....	4
• CPU_ERR_05 Workaround was updated.....	4
• GSC_ERR_01 Category was updated.....	5
• GSC_ERR_01 Module was updated.....	5
• GSC_ERR_01 Function was updated.....	5
• GSC_ERR_01 Description was updated.....	5
• GSC_ERR_01 Workaround was updated.....	5
• IOMUX_ERR_01 Module was updated.....	5
• IOMUX_ERR_01 Function was updated.....	5
• IOMUX_ERR_01 Description was updated.....	5
• IOMUX_ERR_01 Workaround was updated.....	5
• PMCU_ERR_15 Module was updated.....	6
• PMCU_ERR_15 Description was updated.....	6
• PMCU_ERR_15 Workaround was updated.....	6
• PMCU_ERR_15 Function was updated.....	6
• SYSPLL_ERR_01 Workaround was updated.....	6
• TIMER_ERR_06 Module was updated.....	8
• TIMER_ERR_08 Description was updated.....	9
• TIMER_ERR_08 Workaround was updated.....	9
• TIMER_ERR_08 Module was updated.....	9
• TIMER_ERR_08 Function was updated.....	9

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you fully indemnify TI and its representatives against any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to [TI's Terms of Sale](#), [TI's General Quality Guidelines](#), or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products. Unless TI explicitly designates a product as custom or customer-specified, TI products are standard, catalog, general purpose devices.

TI objects to and rejects any additional or different terms you may propose.

Copyright © 2026, Texas Instruments Incorporated

Last updated 10/2025