Errata

MSPM0G3x0x, MSPM0G1x0x Microcontrollers

ABSTRACT

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

Table of Contents

1 Functional Advisories

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

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

Errata Number	Rev B
ADC_ERR_01	✓
ADC_ERR_02	✓
ADC_ERR_04	✓
ADC_ERR_05	✓
ADC_ERR_06	✓
BSL_ERR_01	✓
COMP_ERR_01	✓
COMP_ERR_02	✓
COMP_ERR_03	✓
COMP_ERR_04	✓
CPU_ERR_01	✓
DMA_ERR_01	✓
GPIO_ERR_01	✓
GPIO_ERR_03	✓
I2C_ERR_01	✓
I2C_ERR_02	✓
I2C_ERR_03	✓
I2C_ERR_04	✓
I2C_ERR_05	✓
I2C_ERR_06	✓
PMCU_ERR_06	✓
PMCU_ERR_08	✓
PWREN_ERR_01	✓

Copyright © 2024 Texas Instruments Incorporated

Errata Number	Rev B
RTC_ERR_01	✓
SPI_ERR_01	✓
SPI_ERR_02	✓
SPI_ERR_03	✓
SPI_ERR_04	✓
SPI_ERR_05	✓
SYSOSC_ERR_01	✓
SYSOSC_ERR_02	✓
TIMER_ERR_01	✓
TIMER_ERR_04	✓
TIMER_ERR_06	✓
UART_ERR_01	✓
UART_ERR_02	✓
VREF_ERR_01	✓
VREF_ERR_02	✓
WWDT_ERR_01	✓
WWDT_ERR_02	✓

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."

Copyright © 2024 Texas Instruments Incorporated

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.

Copyright © 2024 Texas Instruments Incorporated

6 Advisory Descriptions

ADC_ERR_01 *ADC Module*

Category

Functional

Function

ADC is unable to trigger fast clock in STANDBY1 mode

Description

When the device is operating in STANDBY1 mode, the ADC module may not correctly assert an asynchronous fast clock request when it is triggered through the event system (for example, when an event publisher such as a timer generates an event to the ADC via the ADCs event subscriber port).

Workaround

Use STANDBY0 or higher power modes when triggering ADC conversions via the event fabric.

ADC_ERR_02 *ADC Module*

Category

Functional

Function

ADC does not release the fast clock request in between periodical triggers

Description

When the ADC is set up in repeat mode and triggered periodically via the event fabric, it does not release its fast clock request in between triggers. IF so, the ADC will hold the clock request AND consume extra power.

Workaround

Configuring the ADC in single mode (run to completion with ADC disabling at end of channel/sequence), AND THEN using a software interrupt at the end of the ADC sequence to re-enable the ADC to wait for the next trigger.

ADC_ERR_04 *ADC Module*

Category

Functional

Function

Increased current in low power mode when ADC is in Repeat mode

Description

When the ADC is configured to Repeat Sequence or Repeat Single mode with EVENT trigger in low power mode (LPM), the ADC module will consume additional current until disabled.

Workaround

Configure ADC to be in Single or Sequence mode (no Repeat), and use an interrupt after conversion to re-enable ADC for next trigger.

Copyright © 2024 Texas Instruments Incorporated

ADC_ERR_05 *ADC Module*

Category

Functional

Function

HW Event generated before enabling IP will stay in queue

Description

When ADC is configured in HW event trigger mode and trigger is generated before enabling ADC will stay in queue. Once ADC is enabled, it will trigger converstion.

Workaround

After configuring ADC in HW trigger mode, enable ADC first before giving external trigger.

ADC_ERR_06 *ADC Module*

Category

Functional

Function

ADC Output code jumps degrading DNL/INL specification

Description

The ADC may have errors at a rate as high as 1 in 2M conversions in 12-bit mode. When a conversion error occurs, it will be a significant random jump in the digital output of the ADC without a corresponding change in the ADC input voltage. The magnitude of this jump is larger near major transitions in the bit values of the ADC result (more bits transitioning from 1->0, or 0->1), and largest around midscale (2048 or 0x800). Depending on the application needs the best workaround may vary, but the following workarounds in software are proposed. Selection of the best workaround is left to the judgment of the system designer.

Workaround

Workaround 1: Upon ADC result outside of application threshold (via ADC Window Comparator or
software thresholding), trigger or wait for another ADC result before making critical system decisions

Workaround 2: During post-processing, discard ADC values which are sufficiently far from the median
or expected value. The expected value should be based on the average of real samples taken in the system, and the threshold for rejection should be based on the magnitude of the measured system noise.

Workaround 3: Use ADC sample averaging to minimize the effect of the results of any single incorrect
conversion.

BSL_ERR_01 *BSL Module*

Category

Functional

Function

Invoking the BSL through application software will fail under certain conditions

Copyright © 2024 Texas Instruments Incorporated

BSL_ERR_01 (continued)

BSL Module

Description

BSL fails to start through invocation within an application (BSL Software Invoke) due to SRAM error code. After a reset, the device will go back to the application instead of invoking BSL due to the error.

This errata does not apply to BSL invocation through hardware invoke methods

Workaround

For applications needing a software invocation of the BSL, clear the entirety of SRAM with assembly code before resetting device. The MSPM0 bsl_software_invoke example within the MSPM0 SDK v 1.20.01.xx or higher contains an example fix within the "bsl_software_invoke" example.

COMP_ERR_01

COMP Module

Category

Functional

Function

Comparator output keeps toggling in STANDBY0 mode

Description

When the comparator inverting input mux (IMSEL) is set to 0 (COMP0_IN0-), and the device operating mode is set to STANDBY0, the comparator output may toggle unexpectedly regardless of the applied input voltages to the comparator.

Workaround

If utilizing comparator in STANDBY0, and applying an external voltage to the negative terminal, utilize channel 1 of the comparator (IMSEL = 1).
If utilizing comparator in STANDBY0, and applying an internal reference voltage (DAC8, VREF, etc.), set IMSEL to a value other than 0 (IMSEL 0).

COMP_ERR_02

COMP Module

Category

Functional

Function

COMP output toggles when DACCODEx is used for hysteresis

Description

IF using the 8-bit DAC within the COMP module as an input to the COMP,
AND DAC output switches between values placed in DACCODEx,
THEN the COMP output toggles as if immediately crossing the new reference point.

This can happen regardless of the setting of the COMPx.CTL2.DACCTL bit.

This is most commonly seen in applications that utilize the two DACCODEx codes for custom or asymmetrical hysteresis for the COMP module.

Copyright © 2024 Texas Instruments Incorporated

COMP_ERR_02 (continued) — ***COMP Module***

Workaround

Utilize the established hysteresis values provided in COMPx.CTL1.HYST register bits.

COMP_ERR_03 — ***COMP Module***

Category

Functional

Function

COMP hysteresis features are non-functional when using input exchange feature

Description

When using hysteresis features of the COMP module, and exchanging the inputs of the COMP (COMPx.CTL1.EXCH = 1), the COMP module becomes unstable.

Workaround

Do not apply internal hysteresis methods when utilizing COMP module in input exchange feature.

COMP_ERR_04 — ***COMP Module***

Category

Functional

Function

Connecting TEMPSENSE as COMP input channel in standby makes COMP output continuously toggle

Description

Temperature sensor is OFF in standby mode. Connecting TEMPSENSE as COMP input channel in standby mode makes COMP output continuously toggle.

Workaround

Do not use TEMSENSE as COMP input in standby mode.

CPU_ERR_01 — ***CPU Module***

Category

Functional

Function

CPU cache content can get corrupted

Description

Cache corruption can occur when switching between accessing Main flash memory, and other memory regions such as NONMAIN or Calibration data areas.

Workaround

Use the following procedure to access areas outside main memory safely: 1. Disable the cache by setting CTL.ICACHE to "0". 2. Perform needed access to memory area. 3. Re-enable cache by setting CTL.ICACHE to "1".

Copyright © 2024 Texas Instruments Incorporated

DMA_ERR_01 *DMA Module*

Category

Functional

Function

DMA or CPU may lose operation when simultaneously accessing peripheral register across clock resource

Description

DMA and CPU are sourced from MCLK. When MCLK is running at a higher frequency than ULPCLK, DMA or CPU may lose operation when simultaneously accessing the peripheral register sourced from ULPCLK, which include all PD0 peripherals. The accessed peripheral or register does not have to be the same. Note: ADC is a PD0 peripheral, while DMA or CPU accessing SVT_MEMRES or SVT_FIFODATA of ADC has no restriction. Example: Assume that DMA access UART0 register, for example DMA write data to TXDATA, if CPU also access PD0 peripheral during DMA operation, for example CPU read data from TIMG0 CTR, then DMA or CPU may lose the data.

Workaround

CPU and DMA should not be accessing PD0 peripherals at the same time when MCLK is running at a higher frequency than ULPCLK.

GPIO_ERR_01 *GPIO Module*

Category

Functional

Function

GPIO wakeup edges may be lost in STANDBY mode

Description

After waking up once through a single GPIO edge, subsequent GPIO wakeup edges can be missed in STANDBY/STOP/SLEEP modes.

Case 1:
STANDBY0 wakeup - IF the MCU is set into STANDBY/STOP/SLEEP mode with an IO in "wake" state AND one sets the IO back to the "non-wake" state for < 3 LFCLK cycles and THEN asserts it again, the next wakeup edge will not be detected.

Case 2:
STANDBY1 wakeup - IF a GPIO edge is used to wakeup AND the GPIO pulse is still active when the device returns to STANDBY1, THEN the device will not detect any subsequent wakeup edges.

Workaround

Case 1:
Ensure that the GPIO is de-asserted while the device is in active mode
OR
Ensure GPIO wakeup pulse is longer than 3 LFCLK cycles

Case 2:
Set GPIO wakeup edge to both falling and rising edges
OR
Ensure GPIO wakeup pulse is not active before entering STANDBY1

Copyright © 2024 Texas Instruments Incorporated

GPIO_ERR_03 *GPIO and DEBUGSS Module*

Category

Functional

Function

On a debugger read to GPIO EVENT0 IIDX, interrupt is cleared.

Description

EVENT0's IIDX of GPIO, on a debugger read is treated as a CPU read and interrupt is getting cleared.

Workaround

During the debug, the IIDX of event0 can be read by software reading RIS.

I2C_ERR_01 *I2C Module*

Category

Functional

Function

I2C module may hold the SDA line in SBMUS mode when an SMBUS quick command is issued

Description

When the I2C module is target mode and configured for SBMUS, IF the bus controller issues an SMBUS quick command addressed to the device (an I2C START condition followed by a 7-bit address, 1-bit R/W signal, 1-bit ACK, and an I2C STOP condition) with the R/W bit set to read, THEN the I2C module may attempt to pull the SDA line low at the same time that the bus controller is attempting to signal the I2C STOP condition, preventing the STOP condition from completing successfully.

Workaround

Load data into the I2C module transmit FIFO with the MSB set to 1 before the address ACK is completed to prevent the I2C module from driving the SDA line low. This will allow the bus controller to issue the STOP condition successfully and complete the SMBUS quick command.

I2C_ERR_02 *I2C Module*

Category

Functional

Function

I2C quick command read mode only works with specific conditions

Description

I2C quick command in read mode works only if TXFIFO has data with MSB set to 1 and clock stretching is disabled.

Workaround

Provide a dummy data in the TXFIFO with MSB set to 1 AND disable the clock stretching (CLKSTRETCH = 0).

Copyright © 2024 Texas Instruments Incorporated

I2C_ERR_03	*I2C Module*
Category	Functional
Function	I2C peripheral mode cannot wake up device when sourced from MFCLK
Description	IF I2C module is configured in peripheral mode AND I2C is clocked from MFCLK (Middle Frequency Clock) AND device is placed in STOP2 or STANDBY0/1 power modes, THEN I2C fails to wakeup the device when receiving data.
Workaround	Set I2C to be clocked by BUSCLK instead of MFCLK, if needing low power wakeup upon receiving data in I2C peripheral mode.

Copyright © 2024 Texas Instruments Incorporated

I2C_ERR_04	*I2C Module*
Category	Functional
Function	When SCL low occurs and target wakeup is enabled, device may clock stretch indefinitely.
Description	When the device is in target mode and SCL is pulled low due to clock stretching or external grounding, if the controller releases the bus, the I2C target is unable to release the clock stretch.
Workaround	Disable the target wakeup enable bit (SWUEN).

I2C_ERR_05	*I2C Module*
Category	Functional
Function	I2C SDA may get stuck to zero if we toggle ACTIVE bit during ongoing transaction
Description	If ACTIVE bit is toggled during an ongoing transfer, its state machine will be reset. However, the SDA and SCL output which is driven by the master will not get reset. There is a situation where SDA is 0 and master has gone into IDLE state, here the master won't be able to move forward from the IDLE state or update the SDA value. Slave's BUSBUSY is set (toggling of the ACTIVE bit is leading to a start being detected on the line) and the BUSBUSY won't be cleared as the master will not be able to drive a STOP to clear it.
Workaround	Do not toggle the ACTIVE bit during an ongoing transaction.

I2C_ERR_06	*I2C Module*
Category	Functional
Function	SMBus High timeout feature fails at I2C clock less than 24KHz onwards
Description	SMBus High timeout feature is failing at I2C clock rate less than 24KHz onwards (20KHz, 10KHz). From SMBUS Spec, the upper limit on SCL high time during active transaction is 50us. Total time taken from writing of START MMR bit to SCL low is 60us, which is >50us. It will trigger the timeout event and let I2C Master goes into IDLE without completing the transaction at the start of transfer itself. Below is detailed explanation. For SCL is configured as 20KHz, SCL low and high period is 30us and 20us respectively. First, START MMR bit write at the same time high timeout counter starts decrementing. Then, it takes one SCL low period (30us) from START MMR bit write to SDA goes low (start condition). Next, it takes another SCL low period (30us) from SDA goes low (start condition) to SCL goes low (data transfer starts) which should stop the high timeout counter at this point. As a total, it takes 60us from counter start to end. However, due to the upper limit(50us) of the high timeout counter, the timeout event will still be triggered although the I2C transaction is working fine without issue.

Copyright © 2024 Texas Instruments Incorporated

I2C_ERR_06 (continued) — *I2C Module*

Workaround

Do not use SMBus High timeout feature when I2C clock less than 24KHz onwards.

PMCU_ERR_06 — *PMCU Module*

Category

Functional

Function

CPU AND DMA are not able to access the flash at the same time

Description

CPU AND DMA cannot concurrently access the flash; for instance, this simultaneous access results in reading incorrect data from the flash during the flash erase operation. The issue can be seen whenever HREADY of the pulled low to CPU due to an ongoing DMA access, program/ erase operation, read Verify/ blank verify operations, basically anything other than CPU.

Workaround

Do not access the flash via CPU AND DMA concurrently. In case of program/ erase operation, read Verify/ blank verify operations, software needs to make sure that CPU does not access flash. This can be ensured by putting code in SRAM while a flash operation is on-going.

PMCU_ERR_08 — *PMCU Module*

Category

Functional

Function

More wakeup time than expected when trigger is given to device while it is transitioning to LPM

Description

If wakeup signal is given to device when it is transitioning to low power mode, an additional wakeup time of approximately 3us will occur.

Workaround

No workaround.

PWREN_ERR_01 — *GPIO Module*

Category

Functional

Function

Peripheral registers are still accessible after disabling PWREN register

Description

When disabling the power of a peripheral by setting the PWREN register to 0, the peripherals registers may appear to retain data values if read. Reading or writing to the registers when PWREN is 0 has no affect as the peripheral has no effect.

The following peripherals are affected: comparator (COMP), operational amplifier (OPA),

Copyright © 2024 Texas Instruments Incorporated

PWREN_ERR_01 (continued)

GPIO Module

TimerA, TimerG, general-purpose input/output (GPIO), and windowed watchdog timer (WWDT), AES and TRNG.

Workaround

When the PWREN register of the peripheral is set to 0, the values of the associated registers should be disregarded or considered invalid.

RTC_ERR_01

RTC Module

Category

Functional

Function

Some RTC Interrupts are not available in STANDBY1

Description

When in STANDBY1, the RTCRDY and RTC_PRESCALER1 interrupts cannot wakeup the device.

Workaround

When waking up the device from STANDBY1 with the RTC, use other available interrupts such as RTC_ALARM and RTC_PRESCALER0.

SPI_ERR_01

SPI Module

Category

Functional

Function

SPI Parity bit is not functional

Description

SPI hardware parity modes are not functional.

Workaround

Do not enable hardware parity via the PTEN or PREN bit in the CTL1 register of the SPI module. Parity computation and checking may be implemented with application software.

SPI_ERR_02

SPI Module

Category

Functional

Function

Missing SPI Clock and data bytes after wake-up from low power mode (LPM)

Description

After device wake-up from a low power state, the SPI module may not properly propagate the first few clock cycles and data bits of the first byte sent out.

Copyright © 2024 Texas Instruments Incorporated

SPI_ERR_02
(continued)

SPI Module

Workaround

To ensure SPI data integrity after a wakeup, use the following sequence when entering and exiting LPMs:

1.Disable SPI module
2.Wait for Interrupt(WFI)- enter LPM
3.Wake up from LPM (any source).
4.Enable the SPI module.

SPI_ERR_03

SPI Module

Category

Functional

Function

When configured as peripheral for a multi-peripheral application, received data will have a right shift

Description

In multi-peripheral scenario, SPI controller first communicates with peripheral0 and then communicates with peripheral1. After finishing communication with peripheral1, the controller again communicates with peripheral0. During the second communication with peripheral0, received data of peripheral0 will have a right shift in the first frame. The peripheral0 is getting first data as 0x3B when the controller sent data 0x76.

Workaround

To support multi peripheral scenario CSCLR needs to be enabled at peripheral end to reset it's RX and TX the bit counters, when there is no active communication happening with that peripheral (CS of that peripheral will be disabled).

SPI_ERR_04

SPI Module

Category

Functional

Function

IDLE/BUSY status toggle after each frame receive when SPI peripheral is in only receive mode.

Description

In case of SPI peripheral in only receiving mode, the IDLE interrupt and
BUSY status are toggling after each frame receive while SPI is receiving data continuously(SPI_PHASE=1). Here there is no data loaded into peripheral(slave) TXFIFO and TXFIFO is empty.

Workaround

Do not use SPI peripheral only receive mode. Set SPI in peripheral(slave) simultaneous transmit and receive mode.

SPI_ERR_05

SPI Module

Category

Functional

Copyright © 2024 Texas Instruments Incorporated

SPI_ERR_05 (continued) ***SPI Module***

Function

SPI Peripheral Receive Timeout interrupt is setting irrespective of RXFIFO data

Description

When using SPI timeout interrupt, the RXTIMEOUT counter started decrementing from the point that peripheral is stopped receiving SPI clock and setting the RXTIMEOUT interrupt irrespective of data exists in RXFIFO or not, which does not match the description in the TRM: SPI peripheral receive timeout(RTOUT) interrupt is "asserted when the receive FIFO is not empty, and no further data is received in the specified time at CTL1.RXTIMEOUT.

Workaround

Repeat load RXTIMEROUT counter value while receive FIFO is empty, and start timeout counting only when receive FIFO gets any data.

SYSOSC_ERR_01 ***SYSOSC Module***

Category

Functional

Function

MFCLK drift when using SYSOSC FCL together with STOP1 mode

Description

When MFCLK is enabled AND SYSOSC is using the frequency correction loop (FCL) mode AND the STOP1 low power operating mode is used, THEN the MFCLK may drift by 2 cycles when SYSOSC shifts from 4MHz back to 32MHz (either upon exit from STOP1 to RUN mode or upon an asynchronous fast clock request that forces SYSOSC to 32MHz).

Workaround

Workaround1: Use STOP0 mode instead of STOP1 mode. There is no MFCLK drift when STOP0 mode is used. Workaround2: Do not use SYSOSC in the FCL mode (leave FCL disabled) when using STOP1.

SYSOSC_ERR_02 ***SYSOSC Module***

Category

Functional

Function

MFCLK does not work when Async clock request is received in an LPM where SYSOSC was disabled in FCL mode

Description

MFCLK will not start to toggle in below scenario:
1. FCL mode is enabled and then MFCLK is enabled
2. Enter a low power mode where SYSOSC is disabled (SLEEP2/STOP2/STANDBY0/ STANDBY1).
3. Now async requst is received from some peripherals which use MFCLK as functional clock.
This is happening because on receiving async request, SYSOSC gets enabled and ulpclk becomes 32MHz. But the SYSOSCTRIM FSM does not move from DISABLE state. Due to this, the FCL bases MFCLK is gated off and it does not toggle at all.

Copyright © 2024 Texas Instruments Incorporated

SYSOSC_ERR_02 (continued)

SYSOSC Module

Workaround

Avoid using this scenario condition.

TIMER_ERR_01

TIMx Module

Category

Functional

Function

Capture mode captures incorrect value when using hardware event to start timer

Description

When using any timer instance in capture mode, starting the timer using a zero (ZCOND) or load (LCOND) condition causes the timer to capture the zero or load value instead of the captured value in the respective TIMx.CC register. This affects periodic use cases such as period and pulse width capture.

Workaround

Use the below software flow to calculate the period or pulse width. See the timx_timer_mode_capture_duty_and_period in the MSPM0-SDK for an example of the workaround.

1.Disable ZCOND or LCOND by setting to 0h.
2.When a capture occurs, the capture value is correctly captured in TIMx.CC
3.Restart the timer by setting TIMx.CTR to the reload value (load or 0).

TIMER_ERR_04

TIMER Module

Category

Functional

Function

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

Description

When using a GPTIMER in one shot mode and CLKDIV.RATIO is not 0, TIMER re-enable may be missed if done close to zero event.

Workaround

TIMER can be disabled first before re-enabling.

TIMER_ERR_06

TIMG Module

Category

Functional

Function

Writing 0 to CLKEN bit does not disable counter

Description

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

Copyright © 2024 Texas Instruments Incorporated

TIMER_ERR_06 (continued)

TIMG Module

Workaround

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

UART_ERR_01

UART Module

Category

Functional

Function

UART start condition not detected when transitioning to STANDBY1 Mode

Description

After servicing an asynchronous fast clock request that was initiated by a UART transmission while the device was in STANDBY1 mode, the device will return to STANDBY1 mode. If another UART transmission begins during the transition back to STANDBY1 mode, the data is not correctly detected and received by the device.

Workaround

Use STANDBY0 mode or higher low power mode when expecting repeated UART start conditions.

UART_ERR_02

UART Module

Category

Functional

Function

UART End of Transmission interrupt not set when only TXE is enabled

Description

UART End Of Transmission (EOT) interrupt does not trigger when the device is set for transmit only (CTL0.TXE = 1, CTL0.RXE = 0). EOT successfully triggers when device is set for transmit and receive (CTL0.TXE = 1, CTL0.RXE = 1)

Workaround

Set both CTL0.TXE and CTL0.RXE bits when utilizing the UART end of transmission interrupt. Note that you do not need to assign a pin as UART receive.

VREF_ERR_01

VREF Module

Category

Functional

Function

VREF READY bit does not get cleared after disabling VREF

Description

The first time the VREF module is enabled after a SYSRST, the VREF READY bit can be used for its intended functionality. If the VREF module is disabled in application, the VREF READY bit does not get cleared. As a result of this errata, subsequent enabling of the VREF module cannot use the VREF READY bit to indicate the VREF module is stable.

Copyright © 2024 Texas Instruments Incorporated

VREF_ERR_01 (continued)

VREF Module

Workaround

When re-enabling the VREF module in application, utilize a TIMER module to wait the maximum VREF startup time, before utilizing the VREF module. Please see the device datasheet for VREF startup time.

VREF_ERR_02

VREF Module

Category

Functional

Function

Switch VREF from 2.5V to 1.4V mode has a very low slew rate

Description

VREF configuration change from 2.5V to 1.4V mode has a very slow slew rate.

Workaround

Use the below procedure to switch VREF mode. 1.Disable VREF by using the ENABLE bit in the CTL0 register before changing configuration to 1.4V mode. 2.Configure pin PA23(VREF+) as GPIO output and drive this pin to logic low for 100us. A 1uF external capacitor is assumed on the VREF+ pin 3.Re-enable VREF in 1.4V by setting the BUGCONFIG bit in the CTL0 register to 1. With this procedure, the time required to switch from 2.5V to 1.4V mode is 200us.

WWDT_ERR_01

WWDT Module

Category

Functional

Function

Watchdog timer 1 (WWDT1) event always does a SYSRST.

Description

WWDT1 event always does a SYSRST irrespective of the SYSTEMCFG.WWDTLP1RSTDIS bit setting. Thusly, WWDT1 cannot trigger "only NMI" events.

Workaround

Use WWDT0 for NMI purposes.

WWDT_ERR_02

WWDT Module

Category

Functional

Function

Window Watchdog Timer 1 (WWDT1) does not create a reset cause

Description

After a reset caused by WWD1, the SYSCTL.RSTCAUSE register does not accurately portray that WWDT1 caused the reset.

Workaround

None.

Copyright © 2024 Texas Instruments Incorporated

7 Revision History

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

Changes from October 31, 2023 to November 30, 2024 (from Revision A (October 2023) to Revision B (December 2024)) **Page**

Copyright © 2024 Texas Instruments Incorporated

Copyright © 2024 Texas Instruments Incorporated

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), 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 will 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 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.

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

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2024, Texas Instruments Incorporated