Stellaris® LM3S8962 RevA2 Errata

This document contains known errata at the time of publication for the Stellaris® LM3S8962 microcontroller. The table below summarizes the errata and lists the affected revisions. See the data sheet for more details.

See also the ARM® Cortex™-M3 errata, ARM publication number PR326-PRDC-009450 v2.0.

Date	Revision	Description
September 2010	2.10	■ Added issue “Hibernation module does not operate correctly” on page 5, replacing previous Hibernation module errata items. ■ Minor edits and clarifications.
July 2010	2.9	■ Added issue “The RTRIS bit in the UARTRIS register is only set when the interrupt is enabled” on page 9.
June 2010	2.8	■ Added issue “External reset does not reset the XTAL to PLL Translation (PLLCFG) register” on page 5.
May 2010	2.7	■ Removed issue "Hibernation Module 4.194304-MHz oscillator supports a limited range of crystal load capacitance values" as it does not apply to this part. ■ Minor edits and clarifications.
April 2010	2.6	■ Removed issue "Writes to Hibernation module registers sometimes fail" as it does not apply to this part. ■ Added issue "Hibernation Module 4.194304-MHz oscillator supports a limited range of crystal load capacitance values." ■ Minor edits and clarifications.
April 2010	2.5	■ Removed issue "Setting Bit 7 in I2C Master Timer Period (I2CMTPR) register may have unexpected results". The data sheet description has changed such that this is no longer necessary. ■ Minor edits and clarifications.
February 2010	2.4	■ Added issue “The General-Purpose Timer match register does not function correctly in 32-bit mode” on page 8. ■ Added issue "Setting Bit 7 in I2C Master Timer Period (I2CMTPR) register may have unexpected results".
Jan 2010	2.3	■ "Hard Fault possible when waking from Sleep or Deep-Sleep modes and Cortex-M3 Debug Access Port (DAP) is enabled" has been removed and the content added to the LM3S8962 data sheet. ■ "Ethernet number of Packets decremented early" has been removed and the content added to the LM3S8962 data sheet.
Dec 2009	2.2	Started tracking revision history.

Erratum Number	Erratum Title	Revision(s) Affected
1.1	JTAG pins do not have internal pull-ups enabled at power-on reset	A1, A2

Erratum Number	Erratum Title	Revision(s) Affected
1.2	JTAG INTEST instruction does not work	A1, A2
2.1	Clock source incorrect when waking up from Deep-Sleep mode in some configurations	A1, A2
2.2	PLL may not function properly at default LDO setting	A1, A2
2.3	I/O buffer 5-V tolerance issue	A1, A2
2.4	PLL Runs Fast When Using a 3.6864-MHz Crystal	A1, A2
2.5	External reset does not reset the XTAL to PLL Translation (PLLCFG) register	A1, A2
3.1	Hibernation module does not operate correctly	A1, A2
4.1	MERASE bit of the FMC register does not erase the entire Flash array	A1, A2
5.1	GPIO input pin latches in the Low state if pad type is open drain	A1, A2
5.2	GPIO pins may glitch during power supply ramp up	A1, A2
6.1	General-purpose timer Edge Count mode count error when timer is disabled	A1, A2
6.2	General-purpose timer 16-bit Edge Count mode does not load reload value	A1, A2
6.3	The General-Purpose Timer match register does not function correctly in 32-bit mode	A1, A2
7.1	Use of "Always" triggering for ADC Sample Sequencer 3 does not work	A1, A2
7.2	Incorrect behavior with timer ADC triggering when another timer is used in 32-bit mode	A2
7.3	ADC hardware averaging produces erroneous results in differential mode	A2
8.1	The RTRIS bit in the UARTRIS register is only set when the interrupt is enabled	A2
9.1	CAN register accesses require software delays	A1, A2
10.1	PWM pulses cannot be smaller than dead-band time	A1, A2
10.2	PWM interrupt clear misses in some instances	A1, A2
10.3	PWM generation is incorrect with extreme duty cycles	A1, A2
10.4	PWMINTEN register bit does not function correctly	A1, A2
10.5	Sync of PWM does not trigger "zero" action	A1, A2
10.6	PWM "zero" action occurs when the PWM module is disabled	A1, A2
11.1	QEI index resets position when index is disabled	A1, A2
11.2	QEI hardware position can be wrong under certain conditions	A1, A2

1 JTAG and Serial Wire Debug

1.1 JTAG pins do not have internal pull-ups enabled at power-on reset

Description:

Following a power-on reset, the JTAG pins $\overline{\text{TRST}}$, TCK, TMS, TDI, and TDO (PB7 and PC[3:0]) do not have internal pull-ups enabled. Consequently, if these pins are not driven from the board, two things may happen:

- The JTAG port may be held in reset and communication with a four-pin JTAG-based debugger may be intermittent or impossible.
- The receivers may draw excess current.

Workaround:

There are a number of workarounds for this problem, varying in complexity and impact:

1. Add external pull-up resistors to all of the affected pins. This workaround solves both issues of JTAG connectivity and current consumption.
2. Add an external pull-up resistor to $\overline{\text{TRST}}$. Firmware should enable the internal pull-ups on the affected pins by setting the appropriate PUE bits of the appropriate **GPIO Pull-Up Select (GPIOPUR)** registers as early in the reset handler as possible. This workaround addresses the issue of JTAG connectivity, but does not address the current consumption other than to limit the affected period (from power-on reset to code execution).
3. Pull-ups on the JTAG pins are unnecessary for code loaded via the SWD interface or via the serial boot loader. Loaded firmware should enable the internal pull-ups on the affected pins by setting the appropriate PUE bits of the appropriate **GPIOPUR** registers as early in the reset handler as possible. This method does not address the current consumption other than to limit the affected period (from power-on reset to code execution).

Silicon Revision Affected:

A1, A2

1.2 JTAG INTEST instruction does not work

Description:

The JTAG INTEST (Boundary Scan) instruction does not properly capture data.

Workaround:

None.

Silicon Revision Affected:

A1, A2

2 System Control

2.1 Clock source incorrect when waking up from Deep-Sleep mode in some configurations

Description:

In some clocking configurations, the core prematurely starts executing code before the main oscillator (MOSC) has stabilized after waking up from Deep-Sleep mode. This situation can cause undesirable behavior for operations that are frequency dependent, such as UART communication.

This issue occurs if the system is configured to run off the main oscillator, with the PLL bypassed and the DS0SCSRC field of the **Deep-Sleep Clock Configuration (DSLPCLKCFG)** register set to use the internal 12-MHz oscillator, 30-KHz internal oscillator, or 32-KHz external oscillator. When the system is triggered to wake up, the core should wait for the main oscillator to stabilize before

starting to execute code. Instead, the core starts executing code while being clocked from the deep-sleep clock source set in the **DSLPCLKCFG** register. When the main oscillator stabilizes, the clock to the core is properly switched to run from the main oscillator.

Workaround:

Run the system off of the main oscillator (MOSC) with the PLL enabled. In this mode, the clocks are switched at the proper time.

If the main oscillator must be used to clock the system without the PLL, a simple wait loop at the beginning of the interrupt handler for the wake-up event should be used to stall the frequency-dependent operation until the main oscillator has stabilized.

Silicon Revision Affected:

A1, A2

2.2 PLL may not function properly at default LDO setting

Description:

In designs that enable and use the PLL module, unstable device behavior may occur with the LDO set at its default of 2.5 volts or below (minimum of 2.25 volts). Designs that do not use the PLL module are not affected.

Workaround:

Prior to enabling the PLL module, it is recommended that the default LDO voltage setting of 2.5 V be adjusted to 2.75 V using the **LDO Power Control (LDOPCTL)** register.

Silicon Revision Affected:

A1, A2

2.3 I/O buffer 5-V tolerance issue

Description:

GPIO buffers are not 5-V tolerant when used in open-drain mode. Pulling up the open-drain pin above 4 V results in high current draw.

Workaround:

When configuring a pin as open drain, limit any pull-up resistor connections to the 3.3-V power rail.

Silicon Revision Affected:

A1, A2

2.4 PLL Runs Fast When Using a 3.6864-MHz Crystal

Description:

If the PLL is enabled, and a 3.6864-MHz crystal is used, the PLL runs 4% fast.

Workaround:

Use a different crystal whose frequency is one of the other allowed crystal frequencies (see the values shown for the XTAL bit in the **RCC** register).

Silicon Revision Affected:

A1, A2

2.5 External reset does not reset the XTAL to PLL Translation (PLLCFG) register

Description:

Performing an external reset (anything but power-on reset) reconfigures the XTAL field in the **Run-Mode Clock Configuration (RCC)** register to the 6 MHz setting, but does not reset the **XTAL to PLL Translation (PLLCFG)** register to the 6 MHz setting.

Consider the following sequence:

1. Performing a power-on reset results in XTAL = 6 MHz and **PLLCFG** = 6 MHz
2. Write an 8 MHz value to the XTAL field results in XTAL = 8 MHz and **PLLCFG** = 8 MHz
3. RST asserted results in XTAL = 6 MHz and **PLLCFG** = 8 MHz

In the last step, **PLLCFG** was not reset to its 6MHz setting. If this step is followed by enabling the PLL to run from an attached 6-MHz crystal, the PLL then operates at 300MHz instead of 400MHz. Subsequently configuring the XTAL field with the 8 MHz setting does not change the setting of **PLLCFG**.

Workaround:

Set XTAL in **PLLCFG** to an incorrect value, and then to the desired value. The second change updates the register correctly. Do not enable the PLL until after the second change.

Silicon Revision Affected:

A1, A2

3 Hibernation Module

3.1 Hibernation module does not operate correctly

Description:

The Hibernation module on this microcontroller does not operate correctly.

Workaround:

This errata item does not apply to many Stellaris devices, including the LM3S1166, LM3S1636, LM3S1969, and LM3S2919. Refer to the Stellaris Product Selector Guide (www.ti.com/stellaris_search) and Errata documents to find an alternative microcontroller that meets the design requirements for your application.

Silicon Revision Affected:

A1, A2

4 Flash Controller

4.1 MERASE bit of the FMC register does not erase the entire Flash array

Description:

The MERASE bit of the **Flash Memory Control (FMC)** register does not erase the entire Flash array. If the contents of the **Flash Memory Address (FMA)** register contain a value less than 0x20000, only the first 128 KB of the Flash array are erased. If bit 17 (value of 0x20000) is set, then only the upper address range of Flash (greater than 128 KB) is erased.

Workaround:

If the entire array must be erased, the following sequence is recommended:

1. Write a value of 0x00000000 to the **FMA** register.
2. Write a value of 0xA4420004 to the **FMC** register, and poll bit 2 until it is cleared.
3. Write a value of 0x00020000 to the **FMA** register.
4. Write a value of 0xA4420004 to the **FMC** register, and poll bit 2 until it is cleared.

The entire array can also be erased by individually erasing all of the pages in the array.

Silicon Revision Affected:

A1, A2

5 GPIO

5.1 GPIO input pin latches in the Low state if pad type is open drain

Description:

GPIO pins function normally if configured as inputs and the open-drain configuration is disabled. If open drain is enabled while the pin is configured as an input using the **GPIO Alternate Function Select (GPIOAFSEL)**, **GPIO Open Drain Select (GPIOODR)**, and **GPIO Direction (GPIODIR)** registers, then the pin latches Low and excessive current (into pin) results if an attempt is made to drive the pin High. The open-drain device is not controllable.

A GPIO pin is not normally configured as open drain and as an input at the same time. A user may want to do this when driving a signal out of a GPIO open-drain pad while configuring the pad as an input to read data on the same pin being driven by an external device. Bit-banging a bidirectional, open-drain bus (for example, I^2C) is an example.

Workaround:

If a user wants to read the state of a GPIO pin on a bidirectional bus that is configured as an open-drain output, the user must first disable the open-drain configuration and then change the direction of the pin to an input. This precaution ensures that the pin is never configured as an input and open drain at the same time.

A second workaround is to use two GPIO pins connected to the same bus signal. The first GPIO pin is configured as an open-drain output, and the second is configured as a standard input. This

way the open-drain output can control the state of the signal and the input pin allows the user to read the state of the signal without causing the latch-up condition.

Silicon Revision Affected:

A1, A2

5.2 GPIO pins may glitch during power supply ramp up

Description:

Upon completing a POR (power on reset) sequence, the GPIO pins default to a tri-stated input condition. However, during the initial ramp up of the external V_{DD} supply from 0.0 V to 3.3 V, the GPIO pins are momentarily configured as output drivers during the time the internal LDO circuit is also ramping up. As a result, a signal glitch may occur on GPIO pins before both the external V_{DD} supply and internal LDO voltages reach their normal operating conditions. This situation can occur when the V_{DD} and LDO voltages ramp up at significantly different rates. The LDO voltage ramp-up time is affected by the load capacitance on the LDO pin, therefore, it is important to keep this load at a nominal 1 µF value as recommended in the data sheet. Adding significant more capacitance loading beyond the specification causes the time delay between the two supply ramp-up times to grow, which possibly increases the severity of the glitching behavior.

Workaround:

Ensuring that the V_{DD} power supply ramp up is a fast as possible helps minimize the potential for GPIO glitches. Follow guidelines for LDO pin capacitive loading documented in the electrical section of the data sheet. System designers must ensure that, during the V_{DD} supply ramp-up time, possible GPIO pin glitches can cause no adverse effects to their systems..

Silicon Revision Affected:

A1, A2

6 General-Purpose Timers

6.1 General-purpose timer Edge Count mode count error when timer is disabled

Description:

When a general-purpose timer is configured for 16-Bit Input Edge Count Mode, the timer (A or B) erroneously decrements by one when the Timer Enable (TnEN) bit in the **GPTM Control (GPTMCTL)** register is cleared (the timer is disabled).

Workaround:

When the general-purpose timer is configured for Edge Count mode and software needs to “stop” the timer, the timer should be reloaded with the current count + 1 and restarted.

Silicon Revision Affected:

A1, A2

6.2 General-purpose timer 16-bit Edge Count mode does not load reload value

Description:

In Edge Count mode, the input events on the CCP pin decrement the counter until the count matches what is in the **GPTM Timern Match (GPTMTnMATCHR)** register. At that point, an interrupt is asserted and then the counter should be reloaded with the original value and counting begins again. However, the reload value is not reloaded into the timer.

Workaround:

Rewrite the **GPTM Timern Interval Load (GPTMTnILR)** register before restarting.

Silicon Revision Affected:

A1, A2

6.3 The General-Purpose Timer match register does not function correctly in 32-bit mode

Description:

The **GPTM Timer A Match (GPTMTAMATCHR)** register triggers a match interrupt when the lower 16 bits match, regardless of the value of the upper 16 bits.

Workaround:

None.

Silicon Revision Affected:

A1, A2

7 ADC

7.1 Use of "Always" triggering for ADC Sample Sequencer 3 does not work

Description:

When using ADC Sample Sequencer 3 (SS3) and configuring the trigger source to "Always" to enable continuous sampling by programming the SS3 Trigger Select field (EM3) in the **ADC Event Multiplexer Select (ADCEMUX)** register to 0xF, the first sample will be captured, but no further samples will be updated to the sequencer FIFO. Interrupts are continuously generated after the first sample and the FIFO status remains empty.

Workaround:

Software must disable and re-enable the sample sequencer to capture another sample.

Silicon Revision Affected:

A1, A2

7.2 Incorrect behavior with timer ADC triggering when another timer is used in 32-bit mode

Description:

When a timer is configured to trigger the ADC and another timer is configured to be a 32-bit periodic or one-shot timer, the ADC is triggered continuously instead of the specified interval.

Workaround:

Do not use a 32-bit periodic or one-shot timer when triggering ADC. If the timer is in 16-bit mode, the ADC trigger works as expected.

Silicon Revision Affected:

A2

7.3 ADC hardware averaging produces erroneous results in differential mode

Description:

The implementation of the ADC averaging circuit does not work correctly when the ADC is sampling in differential mode and the difference between the voltages is approximately 0.0V.

Workaround:

Do not use hardware averaging in differential mode. Instead, use the FIFO to store results and average them in software.

Silicon Revision Affected:

A2

8 UART

8.1 The RTRIS bit in the UARTRIS register is only set when the interrupt is enabled

Description:

The RTRIS (UART Receive Time-Out Raw Interrupt Status) bit in the **UART Raw Interrupt Status (UARTRIS)** register should be set when a receive time-out occurs, regardless of the state of the enable RTIM bit in the **UART Interrupt Mask (UARTIM)** register. However, currently the RTIM bit must be set in order for the RTRIS bit to be set when a receive time-out occurs.

Workaround:

For applications that require polled operation, the RTIM bit can be set while the UART interrupt is disabled in the NVIC using the IntDisable(n) function in the StellarisWare Peripheral Driver Library, where n is 21, 22, or 49 depending whether UART0, UART1 or UART2 is used. With this configuration, software can poll the RTRIS bit, but the interrupt is not reported to the NVIC.

Silicon Revision Affected:

A2

Fixed:

Not yet fixed.

9 CAN

9.1 CAN register accesses require software delays

Description:

Because of a synchronization issue between the processor clock and the 8-MHz CAN clock, both read and write accesses to CAN registers require a software delay in order to ensure proper operation. If this delay is not observed between reads or writes, then register data corruption will occur, causing problems that are difficult to debug. Due to the nature of the synchronization issue, write accesses and read accesses have slightly different issues.

When performing CAN register write accesses, a delay is required between successive writes to any CAN register. The amount of delay required is related to the ratio of the processor clock to the CAN clock. For example, if the processor clock is 4 times greater than the CAN clock, then there must be a 4-processor-cycle gap between successive writes to the CAN controller. However, in the case that the processor clock is less than or equal to the CAN clock, then there are no write access limitations.

When performing CAN register read accesses, a delay is required between the reads of the CAN registers. The difference with read accesses is that all read accesses to CAN registers must perform a double read to receive the correct data. The first read initiates the read request to the CAN controller and the second read access retrieves the data. This sequence cannot be interrupted by another read to the same CAN controller or the data read by the second read access will have invalid data. This means that code that reads the CAN registers must protect this read/delay/read sequence from other asynchronous code, such as interrupt handlers, that access the same CAN controller. Like the case for writing CAN registers, the delay between successive reads to CAN registers is related to the ratio of the processor to the CAN clock. For example, if the processor clock is 4 times greater than the CAN clock, then there must be a 4-cycle gap between reads. However, unlike the write case, when the processor clock is less than or equal to the CAN clock, there still must be a 2-processor cycle delay between read accesses in order to retrieve the correct data. Because this erratum will be fixed in future revisions, software should not take advantage of "pipelining" read operations to help improve access time to the CAN registers. This scheme will not work in future versions of the microcontroller and should be avoided.

Debugger accesses to the CAN registers will also show these issues, usually when debuggers perform read accesses to display the register data in a memory window, or in some cases, a register display window. The data displayed in the memory window will not show the correct data for the CAN registers. In most cases, the read accesses are slow and in sequence so they will show the CAN registers in the memory window offset by one word. However, this cannot be guaranteed as the debugger could possibly read the registers too quickly or not in address order and display invalid data.

Workaround:

In order to safely read or write the CAN registers, delays must be inserted for the correct number of cycles. Writes can delay before or after the CAN register write depending on the system needs, while reads must always perform a double-read to get data back from the CAN register. The Stellaris® Peripheral Driver Library (DriverLib) provides the following two functions to perform the delays necessary for reading or writing the CAN registers: CANReadReg and CANWriteReg. The default behavior is tuned for a 50-MHz processor clock via the define (CAN_RW_DELAY) in the can.c file of DriverLib. If the processor clock is lower, this value can be changed and DriverLib can be rebuilt

for more optimal performance. Care should be taken when adjusting this value as different compilers may generate the looping code differently. When this errata is fixed, future releases of DriverLib will replace these functions with direct hardware accesses to the registers.

As an example, the amount of delay necessary if the processor clock is 25 MHz and the CAN clock is 8 MHz is 3.125 processor clocks or at least 4 processor clocks. When reading CAN registers, no other CAN accesses can occur. This requires protecting the non-interrupt code from interrupt handlers corrupting the read operations. This precaution is not required for writes, as the default interrupt latency is higher than the delay necessary at 50 MHz.

To write a CAN register, use the following simple sequence:

1. Write the CAN register.
2. Delay for (processor clock/CAN clock) processor cycles.

To read a CAN register, use the following simple sequence:

1. Acquire CAN mutex (mutual exclusion).
2. Read the CAN register and discard the data.
3. Delay for (processor clock/CAN clock) processor cycles.
4. Read the CAN register again to get the correct data.
5. Release CAN mutex.

The mutex used to protect CAN access can be done more than one way. One method is to simply disable interrupts for the CAN controller that is being accessed during read accesses. Whatever method is used, it must be sure to protect against any asynchronous code that accesses the same CAN controller as the code that it interrupts.

Silicon Revision Affected:

A1, A2

10 PWM

10.1 PWM pulses cannot be smaller than dead-band time

Description:

The dead-band generator in the PWM module has undesirable effects when receiving input pulses from the PWM generator that are shorter than the dead-band time. For example, providing a 4-clock-wide pulse into the dead-band generator with dead-band times of 20 clocks (for both rising and falling edges) produces a signal on the primary (non-inverted) output that is High except for 40 clocks (the combined rising and falling dead-band times), and the secondary (inverted) output is always Low.

Workaround:

User software must ensure that the input pulse width to the dead-band generator is greater than the dead-band delays.

Silicon Revision Affected:

A1, A2

10.2 PWM interrupt clear misses in some instances

Description:

It is not possible to clear a PWM generator interrupt in the same cycle when another interrupt from the same PWM generator is being asserted. PWM generator interrupts are cleared by writing a 1 to the corresponding bit in the **PWM Interrupt Status and Clear (PWMnISC)** register. If a write to clear the interrupt is missed because another interrupt in that PWM generator is being asserted, the interrupt condition still exists, and the PWM interrupt routine is called again. System problems could result if an interrupt condition was already properly handled the first time, and the software tries to handle it again. Note that even if an interrupt event has not been enabled in the **PWM Interrupt and Trigger Enable (PWMnINTEN)** register, the interrupt is still asserted in the **PWM Raw Interrupt Status (PWMnRIS)** register.

Workaround:

In most instances, performing a double-write to clear the interrupt greatly decreases the chance that the write to clear the interrupt occurs on the same cycle as another interrupt. Because each generator has six possible interrupt events, writing the **PWMnISC** register six times in a row guarantees that the interrupt is cleared. If the period of the PWM is small enough, however, this method may not be practical for the application.

Silicon Revision Affected:

A1, A2

10.3 PWM generation is incorrect with extreme duty cycles

Description:

If a PWM generator is configured for Count-Up/Down mode, and the **PWM Load (PWMnLOAD)** register is set to a value N, setting the compare to a value of 1 or N-1 results in steady state signals instead of a PWM signal. For example, if the user configures PWM0 as follows:

- PWMENABLE = 0x00000001
  - PWM0 Enabled
- PWM0CTL = 0x00000007
  - Debug mode enabled
  - Count-Up/Down mode
  - Generator enabled
- PWM0LOAD = 0x00000063
  - Load is 99 (decimal), so in Count-Up/Down mode the counter counts from zero to 99 and back down to zero (200 clocks per period)
- PWM0GENA = 0x000000b0
  - Output High when the counter matches comparator A while counting up
  - Output Low when the counter matches comparator A while counting down
- PWM0DBCTL = 0x00000000

– Dead-band generator is disabled

If the **PWM0 Compare A (PWM0CMPA)** value is set to 0x00000062 (N-1), PWM0 should output a 2-clock-cycle long High pulse. Instead, the PWM0 output is a constant High value.

If the **PWM0CMPA** value is set to 0x00000001, PWM0 should output a 2-clock-cycle long negative (Low) pulse. Instead, the PWM0 output is a constant Low value.

Workaround:

User software must ensure that when using the PWM Count-Up/Down mode, the compare values must never be 1 or the **PWMnLOAD** value minus one (N-1).

Silicon Revision Affected:

A1, A2

10.4 PWMINTEN register bit does not function correctly

Description:

In the **PWM Interrupt Enable (PWMINTEN)** register, the `IntPWM0` (bit 0) bit does not function correctly and has no effect on the interrupt status to the ARM Cortex-M3 processor. This bit should not be used.

Workaround:

PWM interrupts to the processor should be controlled with the use of the **PWM0-PWM2 Interrupt and Trigger Enable (PWMnINTEN)** registers.

Silicon Revision Affected:

A1, A2

10.5 Sync of PWM does not trigger "zero" action

Description:

If the **PWM Generator Control (PWM0GENA)** register has the `ActZero` field set to 0x2, then the output is set to 0 when the counter reaches 0, as expected. However, if the counter is cleared by setting the appropriate bit in the **PWM Time Base Sync (PWMSYNC)** register, then the "zero" action is not triggered, and the output is not set to 0.

Workaround:

None.

Silicon Revision Affected:

A1, A2

10.6 PWM "zero" action occurs when the PWM module is disabled

Description:

The zero pulse may be asserted when the PWM module is disabled.

Workaround:

None.

Silicon Revision Affected:

A1, A2

11 QEI

11.1 QEI index resets position when index is disabled

Description:

When the QEI module is configured to not reset the position on detection of the index signal (that is, the ResMode bit in the **QEI Control (QEICTL)** register is 0), the module resets the position when the index pulse occurs. The position counter should only be reset when it reaches the maximum value set in the **QEI Maximum Position (QEIMAXPOS)** register.

Workaround:

Do not rely on software to disable the index pulse. Do not connect the index pulse if it is not needed.

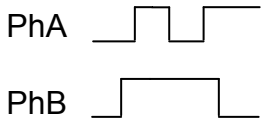
Silicon Revision Affected:

A1, A2

11.2 QEI hardware position can be wrong under certain conditions

Description:

The **QEI Position (QEIPOS)** register can be incorrect if the QEI is configured for quadrature phase mode (SigMode bit in **QEICTL** register = 0) and to update the position counter of every edge of both PhA and PhB (CapMode bit in **QEICTL** register = 1). This error can occur if the encoder is stepped in the reverse direction, stepped forward once, and then continues in the reverse direction. The following sequence of transitions on the PhA and PhB pins causes the error:



Assuming the starting position prior to the above PhA and PhB sequence is 0, the position after the falling edge on PhB should be -3, however the **QEIPOS** register will show the position to be -1.

Workaround:

Configure the QEI to update the position counter on every edge on PhA only (CapMode bit in **QEICTL** register = 0). The effective resolution is reduced by 50%. If full resolution position detection is required by updating the position counter on every edge of both PhA and PhB, no workaround is available. Hardware and software must take this into account.

Silicon Revision Affected:

A1, A2

Texas Instruments Incorporated
108 Wild Basin, Suite 350
Austin, TX 78746
http://www.ti.com/stellaris

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Communications and Telecom	www.ti.com/communications
DSP	dsp.ti.com	Computers and Peripherals	www.ti.com/computers
Clocks and Timers	www.ti.com/clocks	Consumer Electronics	www.ti.com/consumer-apps
Interface	interface.ti.com	Energy	www.ti.com/energy
Logic	logic.ti.com	Industrial	www.ti.com/industrial
Power Mgmt	power.ti.com	Medical	www.ti.com/medical
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Space, Avionics & Defense	www.ti.com/space-avionics-defense
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video and Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless-apps

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265