Stellaris® LM3S2110 RevA2 Errata

This document contains known errata at the time of publication for the Stellaris LM3S2110 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.

Table 1. Revision History

Date	Revision	Description
August 2011	3.0	■ Added issue “Standard R-C network cannot be used on $\overline{\text{RST}}$ to extend POR timing” on page 5. ■ Added issue “Writes to Hibernation module registers may change the value of the RTC” on page 8. ■ Clarified issue “General-purpose timer 16-bit Edge Count or Edge Time mode does not load reload value” on page 10 to include Edge-Time mode.
September 2010	2.10	■ 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 11.
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	■ Minor edits and clarifications.
April 2010	2.6	■ 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 11. ■ 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 LM3S2110 data sheet.
Dec 2009	2.2	Started tracking revision history.

Table 2. List of Errata

Erratum Number	Erratum Title	Module Affected	Revision(s) Affected
1.1	JTAG pins do not have internal pull-ups enabled at power-on reset	JTAG and Serial Wire Debug	A2
1.2	JTAG INTEST instruction does not work	JTAG and Serial Wire Debug	A2

Erratum Number	Erratum Title	Module Affected	Revision(s) Affected
2.1	Clock source incorrect when waking up from Deep-Sleep mode in some configurations	System Control	A2
2.2	PLL may not function properly at default LDO setting	System Control	A2
2.3	I/O buffer 5-V tolerance issue	System Control	A2
2.4	PLL Runs Fast When Using a 3.6864-MHz Crystal	System Control	A2
2.5	External reset does not reset the XTAL to PLL Translation (PLLCFG) register	System Control	A2
2.6	Standard R-C network cannot be used on $\overline{\text{RST}}$ to extend POR timing	System Control	A2
3.1	Hibernation module low VBAT detection does not work as expected	Hibernation Module	A2
3.2	Performing a system-wide reset also resets the Hibernation module and all of its registers	Hibernation Module	A2
3.3	Hibernation module may have higher current draw than specified in data sheet under certain conditions	Hibernation Module	A2
3.4	Hibernation module returns from the Hibernation state to the Wake state regardless of the status of the VDD supply to the microcontroller	Hibernation Module	A2
3.5	Certain Hibernation module register writes cause RTC Counter register inaccuracy	Hibernation Module	A2
3.6	Hibernation module state retention registers may corrupt after Wake sequence	Hibernation Module	A2
3.7	Writes to Hibernation module registers may change the value of the RTC	Hibernation Module	A2
4.1	GPIO input pin latches in the Low state if pad type is open drain	GPIO	A2
4.2	GPIO pins may glitch during power supply ramp up	GPIO	A2
5.1	General-purpose timer Edge Count mode count error when timer is disabled	General-Purpose Timers	A2
5.2	General-purpose timer 16-bit Edge Count or Edge Time mode does not load reload value	General-Purpose Timers	A2
5.3	The General-Purpose Timer match register does not function correctly in 32-bit mode	General-Purpose Timers	A2
6.1	The RTRIS bit in the UARTRIS register is only set when the interrupt is enabled	UART	A2
7.1	CAN register accesses require software delays	CAN	A2
8.1	PWM pulses cannot be smaller than dead-band time	PWM	A2
8.2	PWM interrupt clear misses in some instances	PWM	A2
8.3	PWM generation is incorrect with extreme duty cycles	PWM	A2
8.4	PWMINTEN register bit does not function correctly	PWM	A2
8.5	Sync of PWM does not trigger "zero" action	PWM	A2
8.6	PWM "zero" action occurs when the PWM module is disabled	PWM	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 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 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:

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:

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:

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:

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:

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:

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. $\overline{\text{RST}}$ 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:

A2

2.6 Standard R-C network cannot be used on $\overline{\text{RST}}$ to extend POR timing

Description:

The standard R-C network on $\overline{\text{RST}}$ does not work to extend POR timing beyond the 10 ms on-chip POR. Instead of following the standard capacitor charging curve, $\overline{\text{RST}}$ jumps straight to 3 V at power

on. The capacitor is fully charged by current out of the $\overline{\text{RST}}$ pin and does not extend or filter the power-on condition. As a result, the reset input is not extended beyond the POR.

Workaround:

Add a diode to block the output current from $\overline{\text{RST}}$. This helps to extend the $\overline{\text{RST}}$ pulse, but also means that the R-C is not as effective as a noise filter.

Silicon Revision Affected:

A2

3 Hibernation Module

3.1 Hibernation module low VBAT detection does not work as expected

Description:

The Hibernation module is designed to detect a low V_{BAT} condition. This feature is enabled by setting the LOWBATEN bit in the **Hibernation Control (HIBCTL)** register. When enabled, the low V_{BAT} detection incorrectly detects a low V_{BAT} condition at 3.15 V. This is supposed to trigger a low V_{BAT} condition at 2.35 V.

Workaround:

This feature should not be used and must be disabled by clearing the LOWBATEN bit in the **HIBCTL** register. The battery voltage can be sensed using a hardware workaround by wiring an analog comparator input pin to the battery and setting the comparator reference pin voltage to the desired detection trip level.

Silicon Revision Affected:

A2

3.2 Performing a system-wide reset also resets the Hibernation module and all of its registers

Description:

Performing a system-wide reset by asserting the $\overline{\text{RST}}$ pin, encountering a Brown-Out Reset, or setting the SYSRESETREQ bit in the ARM Cortex-M3 Application Interrupt and Reset Control register also incorrectly causes the Hibernation module to be reset. All of the Hibernation module registers are reset, including the battery-backed **Hibernation Data (HIBDATA)** and the **Hibernation RTC Counter (HIBRTCC)** registers.

Workaround:

None.

Silicon Revision Affected:

A2

3.3 Hibernation module may have higher current draw than specified in data sheet under certain conditions

Description:

If a battery voltage is applied to the VBAT power pin prior to power being applied to the VDD power pins of the device, the current draw from the VBAT pin is greater than expected. The current may be as high as 1.6 mA instead of the data sheet specified 17 µA. The condition exists until power is applied to the VDD pin. Once the VDD pin has been powered, the VBAT current draw functions as expected. The VDD pin can then be powered up and down as required and the VBAT pin current specification is maintained.

Workaround:

The VBAT pin higher-than-specified current draw condition can be avoided if the microcontroller's VDD power pins are powered on prior to the time a battery voltage is initially applied to the VBAT pin.

Silicon Revision Affected:

A2

3.4 Hibernation module returns from the Hibernation state to the Wake state regardless of the status of the VDD supply to the microcontroller

Description:

When the Hibernation module is in Hibernation mode and receives an event to initiate a wake-up (assertion of the $\overline{\text{WAKE}}$ pin, RTC match, or low-battery detect), the $\overline{\text{HIB}}$ signal is de-asserted, enabling the external regulator to provide V_{DD} to the device. The device then performs a Power-On Reset (POR) and the software can query the **Hibernation Raw Interrupt Status (HIBRIS)** register to determine if a hibernation wake occurred.

If the regulator does not have a power source, or for some reason V_{DD} is not immediately applied to the device when the $\overline{\text{HIB}}$ signal is de-asserted, the Hibernation module still exits from its Hibernation state to its Wake state, but the device is not able to perform its POR sequence. If power is then restored to the regulator (providing V_{DD} to the device), the device executes a POR, however, the state of the Hibernation module is incorrectly reset, and all of the registers within the Hibernation module, including the **Hibernation RTC Counter (HIBRTCC)** and **Hibernation Data (HIBDATA)** registers are set to their reset state.

Workaround:

Always ensure that power is available to the regulator controlled by the $\overline{\text{HIB}}$ signal when the $\overline{\text{HIB}}$ signal is de-asserted during a wake event.

Silicon Revision Affected:

A2

3.5 Certain Hibernation module register writes cause RTC Counter register inaccuracy

Description:

The Hibernation module contains a **Hibernation RTC Counter (HIBRTCC)** register that keeps an accumulated running one-second count. This register is updated from an internal, 32-KHz counter which is not visible to the user. As this counter value rolls over, it provides a tick count once per second to the **HIBRTCC** register. Register writes to the **Hibernation RTC Match (HIBRTCM0, HIBRTCM1)** registers, the **Hibernation RTC Trim (HIBRTCT)** register, or the **Hibernation Data (HIBDATA)** register cause the 32-KHz counter to reset to its start value. The result is that the **HIBRTCC** register value loses accuracy each time one of these Hibernation registers is written because the 32-KHz counter is reset in the midst of its one-second count window, effectively delaying the **HIBRTCC** register counter value from its ideal value.

Workaround:

To minimize the amount of introduced error, the application can write to the affected registers immediately after the RTC rolls over. The RTC rollover can be detected by polling the RTC value in the **HIBRTCC** register and waiting for it to change. If the application writes to the affected registers with a predictable and repeatable pattern, then the **HIBRTCT** register can be used to compensate for introduced error.

Silicon Revision Affected:

A2

3.6 Hibernation module state retention registers may corrupt after Wake sequence

Description:

When transitioning from a Hibernate condition to a Wake-up state, the **Hibernation Control (HIBCTL)** register and **Hibernation Interrupt Mask (HIBIM)** register values may occasionally become corrupted. This situation occurs after a wake-up event caused by an RTC match condition or an external signal asserting the $\overline{\text{WAKE}}$ pin of the device.

Workaround:

Software should mirror the data in the **HIBCTL** and **HIBIM** registers to the **Hibernation Data (HIBDATA)** register before initiating a Hibernation sequence. Immediately upon returning from the Hibernation state to the Wake-up state, software should write the data mirrored from the HIBDATA registers back into the **HIBCTL** and **HIBIM** registers.

Silicon Revision Affected:

A2

3.7 Writes to Hibernation module registers may change the value of the RTC

Description:

If the Hibernation module's RTC counter is active, any write to certain Hibernation module registers that occurs while the RTC counter is changing from the current value to the next can cause corruption of the RTC counter stored in the **HIBRTCC** register. Registers affected are: **HIBRTCC**, **HIBRTCM0**, **HIBRTCM1**, **HIBRTCLD**, **HIBRTCT**, and **HIBDATA**.

Workaround:

The user application must guarantee that writes to the affected Hibernation module registers cannot occur on the RTC counter boundary. Any initial configuration of the affected Hibernation module registers must be done before enabling the RTC counter.

There are two ways to update affected Hibernation Module registers after initial configuration:

1. Use the Hibernation RTC match interrupt to perform writes to the affected Hibernation module registers. Assuming the interrupt is guaranteed to be serviced within 1 second, this technique provides a mechanism for the application to know that the RTC update event has occurred and that it is safe to write data to the affected Hibernation module registers. This method is useful for applications that don't require many writes to Hibernation module registers.

2. Set up a secondary time-keeping resource to indicate when it is safe to perform writes to the affected Hibernation module registers. For example, use a general purpose timer in combination with the Hibernation RTC match interrupt. In this scenario, the RTC match interrupt is used to both update the match register value and enable the general purpose timer in one-shot mode. The timer must be configured to have a maximum time-out period of less than 1 second. In this configuration, a global variable is used to indicate that it is safe to perform writes to the affected Hibernation module registers. When the one-shot timer times out, the timer interrupt updates the global variable to indicate that writes are no longer safe. This procedure is repeated on every RTC match interrupt.

Silicon Revision Affected:

A2

4 GPIO

4.1 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:

A2

4.2 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:

A2

5 General-Purpose Timers

5.1 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:

A2

5.2 General-purpose timer 16-bit Edge Count or Edge Time mode does not load reload value

Description:

In Edge Count or Edge Time 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:

A2

5.3 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:

A2

6 UART

6.1 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

7 CAN

7.1 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:

A2

8 PWM

8.1 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:

A2

8.2 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:

A2

8.3 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:

A2

8.4 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:

A2

8.5 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:

A2

8.6 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:

A2

Texas Instruments Incorporated
108 Wild Basin, Suite 350
Austin, TX 78746
http://www.ti.com/stellaris
http://www-k.ext.ti.com/sc/technical-support/product-information-centers.htm

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**	
Audio	www.ti.com/audio	Communications and Telecom	www.ti.com/communications
Amplifiers	amplifier.ti.com	Computers and Peripherals	www.ti.com/computers
Data Converters	dataconverter.ti.com	Consumer Electronics	www.ti.com/consumer-apps
DLP® Products	www.dlp.com	Energy and Lighting	www.ti.com/energy
DSP	dsp.ti.com	Industrial	www.ti.com/industrial
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Security	www.ti.com/security
Logic	logic.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Power Mgmt	power.ti.com	Transportation and Automotive	www.ti.com/automotive
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Mobile Processors	www.ti.com/omap		
Wireless Connctivity	www.ti.com/wirelessconnectivity		
	TI E2E Community Home Page		e2e.ti.com

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