TMS320R2811 and TMS320R2812

Digital Signal Processors

Silicon Errata

SPRZ226C
June 2004 − Revised March 2009

TEXAS INSTRUMENTS

Contents

1 Introduction

This document describes the silicon updates to the functional specifications for the TMS320F2811 and TMS320F2812 digital signal processors. The updates are applicable to:

- TMS320R2812 (179-ball MicroStar BGA™, GHH suffix)
- TMS320R2812 (179-ball MicroStar BGA™, ZHH suffix)
- TMS320R2812 (176-pin low-profile quad flatpack [LQFP], PGF suffix)
- TMS320R2811 (128-pin LQFP, PBK suffix)

1.1 Quality and Reliability Conditions

TMX Definition

Texas Instruments (TI) does not warranty either (1) electrical performance to specification, or (2) product reliability for products classified as “TMX.” By definition, the product has not completed data sheet verification or reliability performance qualification according to TI Quality Systems Specifications.

The mere fact that a “TMX” device was tested over a particular temperature range and voltage range should not, in any way, be construed as a warranty of performance.

TMP Definition

TI does not warranty product reliability for products classified as “TMP.” By definition, the product has not completed reliability performance qualification according to TI Quality Systems Specifications; however, products are tested to a published electrical and mechanical specification.

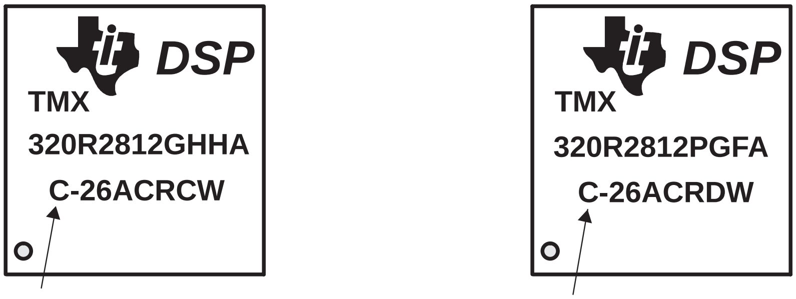
TMS Definition

Fully-qualified production device

MicroStar BGA is a trademark of Texas Instruments.
Other trademarks are the property of their respective owners.

1.2 Revision Identification

The device revision can be determined by the lot trace code marked on the top of the package. The locations of the lot trace codes for the GHH and PGF packages are shown in Figure 1. Table 1 shows how to determine the silicon revision from the lot trace code.



Lot trace code with second letter blank

Lot trace code with second letter blank

Figure 1. Example, Lot Trace Code for TMX320R2812 (GHH and PGF)

Table 1. Determining Silicon Revision From Lot Trace Code

Second Letter in Prefix of Lot Trace Code	Silicon Revision	Revision ID (0x0883)	Comments
Blank (no second letter in prefix)	Indicates Revision 0	0x0000	This silicon revision is available as TMX only.
A	Indicates Revision A	0x0001	This silicon revision is available as TMS only.
B	Indicates Revision B	0x0002	This silicon revision is available as TMS only.

2 Known Design Marginality/Exceptions to Functional Specifications

2.1 Advisories

Advisory	*Memory: Prefetching Beyond Valid Memory*

Revision(s) Affected: 0, A, B

Details: The C28x CPU prefetches instructions beyond those currently active in its pipeline. If the prefetch occurs past the end of valid memory, then the CPU may receive an invalid opcode.

Workaround: The prefetch queue is 8x16 words in depth. Therefore, code should not come within 8 words of the end of valid memory. This restriction applies to all memory regions (XINTF) and all memory types (SARAM) on the device. Prefetching across the boundary between two valid memory blocks is ok.

Example 1: M1 ends at address 0x7FF and is not followed by another memory block. Code in M1 should be stored no farther than address 0x7F7. Addresses 0x7F8−0x7FF should not be used for code.

Example 2: M0 ends at address 0x3FF and valid memory (M1) follows it. Code in M0 can be stored up to and including address 0x3FF. Code can also cross into M1 up to and including address 0x7F7.

Advisory — *XINTF: XBANK Does Not Properly Extend an Access*

Revision(s) Affected: 0, A, B

Details: When XTIMCLK is not equal to SYSCLKOUT the XBANK logic may not properly delay a pending access. This occurs for some combinations of XINTF zone wait states and XBANK delay cycles. There are two cases when this occurs.

Case 1: When XTIMCLK = 1/2 SYSCLKOUT and XCLKOUT = XTIMCLK

A pending access may not be delayed by the XBANK logic if either:

- WLEAD + WACTIVE + WTRAIL <= XBANK[BCYC] or
- RLEAD + RACTIVE + RTRAIL <= XBANK[BCYC]

Where WLEAD, WACTIVE, WTRAIL, RLEAD, RACTIVE, RTRAIL are defined as shown in Table 2.

Table 2. Pending Access Relationships

	X2TIMING = 0	**X2TIMING = 1**
WLEAD	XTIMING x [XWRLEAD]	XTIMING x [XWRLEAD] x 2
WACTIVE	XTIMING x [XWRACTIVE] + 1	XTIMING x [XWRACTIVE] x 2 + 1
WTRAIL	XTIMING x [XWRTRAIL]	XTIMING x [XWRTRAIL] x 2
RLEAD	XTIMING x [XRDLEAD]	XTIMING x [XRDLEAD] x 2
RACTIVE	XTIMING x [XRDACTIVE] + 1	XTIMING x [XRDACTIVE] x 2 + 1
RTRAIL	XTIMING x [XRDTRAIL]	XTIMING x [XRDTRAIL] x 2

In Table 2, XTIMINGx refers to the XTIMING register for Zone x. When XBANK delay cycles are added between two accesses, Zone x refers to the first zone in the sequence. For example: if XBANK[BANK] = 7, then delay cycles will be added to any access into or out of Zone 7. This means:

- Access to Zone 0 followed by Zone 7: the timing of Zone 0 is critical.
- Access to Zone 1 followed by Zone 7: the timing of Zone 1 is critical.
- Access to Zone 7 followed by Zone 0: the timing of Zone 7 is critical.

Thus, the timing of any zone involved in bank switching must be considered.

Case 2) When XTIMCLK = 1/2 SYSCLKOUT and XCLKOUT = 1/2 XTIMCLK:

A pending access may not be delayed properly by the XBANK logic if XBANK[BCYC] = 4 or XBANK[BCYC] = 6.

XINTF: XBANK Does Not Properly Extend an Access (continued)

Workaround(s):

1. **Case 1) If XTIMCLK = 1/2 SYSCLKOUT and XCLKOUT = XTIMCLK then select:**

   XBANK[BCYC] <= WLEAD + WACTIVE + WTRAIL   and

   XBANK[BCYC] <= RLEAD + RACTIVE + RTRAIL

When XBANK delay cycles are added between two accesses, the timing restriction applies to the first zone accessed as described earlier. The timing of any zone involved in bank switching must be considered.

Table 3 shows examples of valid XBANK[BCYC]selections. This list is not exhaustive.

Table 3. Examples of Valid XBANK Selections

XWRLEAD XRDLEAD	WRACTIVE XRDACTIVE	XWRTRAIL XRDTRAIL	X2TIMING	WLEAD + WACTIVE + WTRAIL	Choose XBANK[BCYC]
1	2	1	0	5	< 5
1	3	1	0	6	< 6
2	3	1	0	7	< 7
1	0	1	1	3	< 3
1	1	0	1	5	< 5
1	1	1	1	7	< 7

2. **Case 2: If XTIMCLK = 1/2 SYSCLKOUT and XCLKOUT = 1/2 XTIMCLK, then select:**

   XBANK[BCYC] != 4 and

   XBANK[BCYC] != 6

Advisory — *SCI: SCI Bootloader Does Not Clear the ABD Bit Before Auto-Baud Lock*

Revision(s) Affected: 0, A, B

Details: The SCI ROM bootloader code does not correctly clear the Auto-Baud Detect (ABD) bit in the SCIFFCT register before the auto-baud process begins. The bootloader code fragment is shown below:

```
// Prepare for autobaud detection
// Set the CDC bit to enable autobaud detection
// and clear the ABD bit
SCIARegs.SCIFFCT.all = 0x2000;
```

The comments incorrectly state that the ABD bit is cleared. The ABD bit is cleared by writing a 1 to the ABD_CLR bit (bit 14) of the SCIFFCT register. This situation does not hinder operation from power up or reset because the ABD bit is cleared by default after reset. If, however, the bootloader is invoked a second time from software, then the ABD bit will not be cleared and autobaud lock will not occur properly.

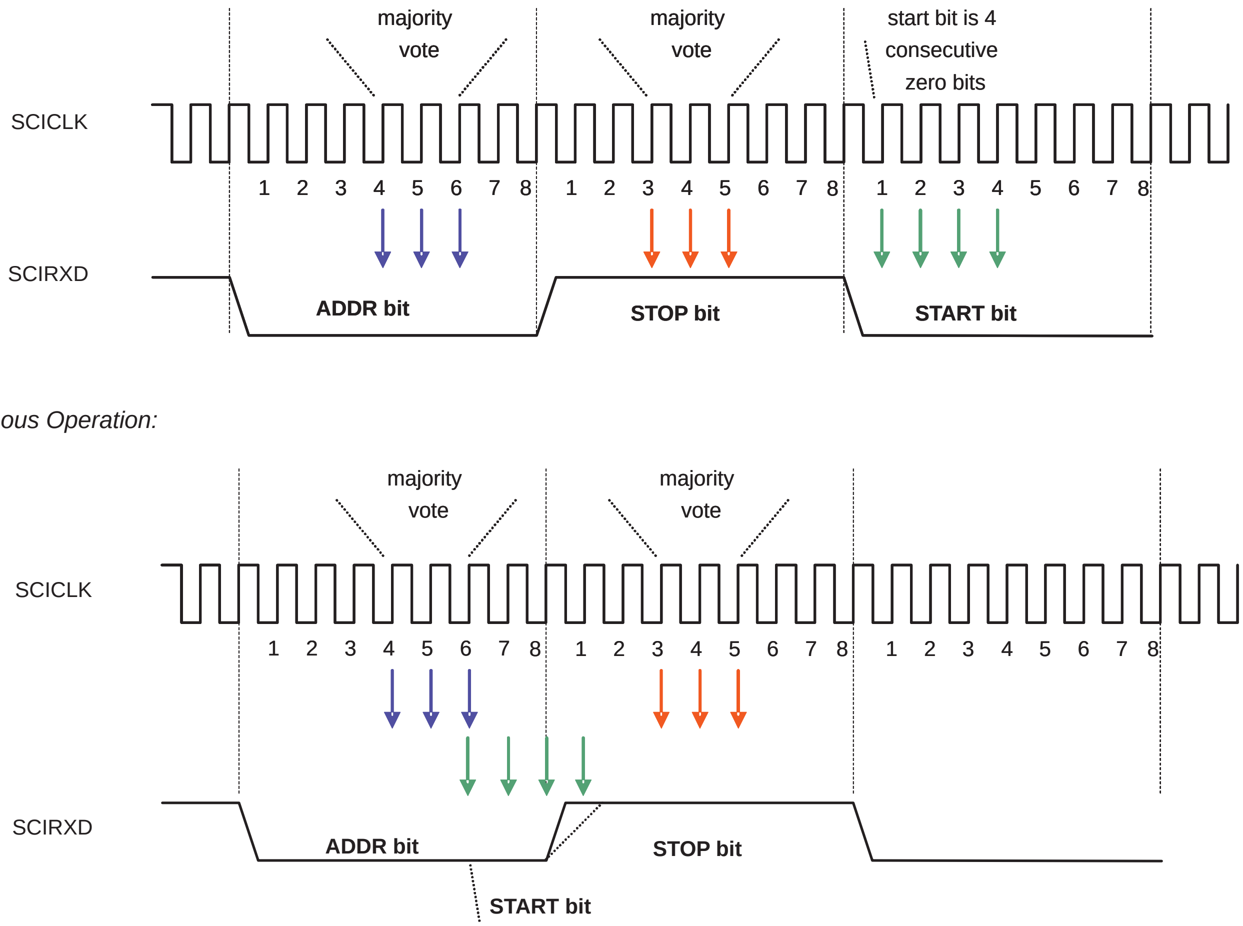
Workaround: If the bootloader is going to be re-invoked by software, the user’s code must first clear the ABD bit before calling the bootloader. To do this, write a 1 to the ABD CLR bit (bit 14) in the SCIFFCT register.

Advisory *SCI: Incorrect Operation of SCI in Address Bit Mode*

Revision(s) Affected: 0, A, B

Details: SCI does not look for STOP bit after the ADDR bit. Instead, SCI starts looking for the start bit beginning on sub-sample 6 of the ADDR bit. Slow rise time from ADDR to STOP bit can cause false START bit to occur since fourth sub-sample for the start bit may be sensed low.

Expected Operation:



Erroneous Operation:

Figure 2. Difference Between Expected and Erroneous Operation of START Bit

Workaround: Program the baud rate of the SCI to be slightly slower than the actual. This will cause the fourth sub-sample of the false START bit to be delayed in time, and therefore occur more towards the middle of the STOP bit (away from the signal transition region). The amount of baud slowing needed depends on the rise time of the signal in the system. Alternatively, IDLE mode of the SCI module may be used, if applicable.

Advisory	*SCI: Bootloader Does Not Clear the ABD Bit After Auto-Baud Lock*

Revision(s) Affected: 0, A, B

Details: The SCI ROM bootloader code does not clear the auto-baud detect (ABD) bit in the SCIFFCT register after the auto-baud process completes. If the SCI-A port is used after the bootloader is executed, transmit interrupts (SCITXINTA) will not be able to occur, nor will the auto-baud lock feature of SCI-A work correctly.

Workaround: If the SCI bootloader has been executed, the user’s application code should clear the ABD bit by writing a 1 to ABD CLR (bit 14) in the SCIFFCT register before enabling the SCITXINTA interrupt, and before using the auto-baud feature.

Advisory *eCAN: CPU Access to the eCAN Registers May Fail If It Is in Conflict With an eCAN Access to the eCAN Registers*

Revision(s) Affected: 0, A, B

Details: If contention exists between the CPU and the eCAN controller for access to certain eCAN register areas, a CPU read may erroneously read all zeros (0x00000000), and a CPU write may erroneously fail to execute. Specifically:

1) If the CPU reads the eCAN mailbox RAM area (MSGID, MSGCTRL, MDL, or MDH registers) at the same time that the eCAN controller is accessing (reading or writing) the LAM/MOTO/MOTS register area, the CPU may erroneously read all zeros (0x00000000).

Workaround: For all CPU reads from the eCAN mailbox RAM area, check to see if the read returns all zeros. If so, the CPU should perform a second read. If the second read returns zero as well, then the data is correctly zero. If the second read returns a non−zero value, then the second data is the correct value. Note that interrupts must be disabled during the consecutive CPU reads. A C−code example is as follows:

```
asm(" PUSH ST1") ;                   // save ST1
asm(" SETC INTM");                   // disable interrupts
 // Insert first CPU read here
 // Insert second CPU read here
asm(" POP ST1");                     // restore ST1
```

Details: 2) If the CPU writes to the eCAN mailbox RAM area (MSGID, MSGCTRL, MDL, or MDH register) at the same time that the eCAN controller is accessing (reading or writing) the LAM/MOTO/MOTS register area, the CPU write may fail to execute.

Workaround: For all CPU writes to the eCAN mailbox RAM area, the CPU should write the data twice. Note that interrupts must be disabled during the consecutive CPU writes. A C−code example is as follows:

```
asm(" PUSH ST1") ;                   // save ST1
asm(" SETC INTM");                   // disable interrupts
 // Insert first CPU write here
 // Insert second CPU write here
asm(" POP ST1");                     // restore ST1
```

Details: 3) If the CPU reads the LAM/MOTO/MOTS register area at the same that the eCAN controller is accessing (reading or writing) the eCAN mailbox RAM area (MSGID, MSGCTRL, MDL, or MDH registers), the CPU may erroneously read all zeros (0x00000000).

eCAN: CPU Access to the eCAN Registers May Fail If It Is in Conflict With an eCAN Access to the eCAN Registers (Continued)

Workaround: For all CPU reads from the LAM/MOTO/MOTS register area, check to see if the read returns all zeros. If so, the CPU should perform a second read. If the second read returns zero as well, then the data is correctly zero. If the second read returns a non−zero value, then the second data is the correct value. Note that interrupts must be disabled during the consecutive CPU reads A C−code example is as follows:

```
asm(" PUSH ST1") ;                  // save ST1
asm(" SETC INTM");                  // disable interrupts
 // Insert first CPU read here
 // Insert second CPU read here
asm(" POP ST1");                    // restore ST1
```

Details: 4) If the CPU writes to the LAM/MOTO/MOTS register area at the same that the eCAN controller is accessing (reading or writing) the eCAN mailbox RAM area (MSGID, MSGCTRL, MDL, or MDH registers), the CPU write may fail to execute.

Workaround: For all CPU writes to the LAM/MOTO/MOTS register area, the CPU should write the data twice with a minimum of 4 CPU cycles in between the writes. Note that interrupts must be disabled during the consecutive CPU writes. A C−code example is as follows:

```
asm(" PUSH ST1") ;                  // save ST1
asm(" SETC INTM");                  // disable interrupts
 // Insert first CPU write here
asm(" RPT #2 || NOP");              // wait 4 CPU cycles
// Insert second CPU write here
asm(" POP ST1");                    // restore ST1
```

NOTES: 1. An example of the eCAN controller reading the LAM/MOTO/MOTS register area is a read of the LAMn register to check if a received message passes the acceptance mask filtering criterion. This happens during reception of a frame.

2. An example of the eCAN controller writing to the LAM/MOTO/MOTS register area is a write to the MOTSn register to update the time−stamp upon successful transmission of a frame.

3. An example for the eCAN controller attempting to read the mailbox RAM area (MSGID, MSGCTRL, MDL & MDH registers) is right before transmission.

4. An example for the eCAN controller attempting to write to the mailbox RAM area (MSGID, MSGCTRL, MDL & MDH registers) is right after reception.

Advisory — *eCAN: Abort Acknowledge Bit Not Set*

Revision(s) Affected: 0, A, B

Details: After setting a transmission request reset (TRR) register bit to abort a message, there are some rare instances where the TRRn and TRSn bits will clear without setting the abort acknowledge (AAn) bit. The transmission itself is correctly aborted, but no interrupt is asserted and there is no indication of a pending operation.

In order for this rare condition to occur, all of the following conditions must happen:

1. The previous message was not successful, either because of lost arbitration or because no node on the bus was able to acknowledge it or because an error frame resulted from the transmission. The previous message need not be from the same mailbox in which a transmit abort is currently being attempted.
2. The TRRn bit of the mailbox should be set in a CPU cycle immediately following the cycle in which the TRSn bit was set. The TRSn bit remaining set due to incompletion of transmission satisfies this condition as well. i.e., the TRSn bit could have been set in the past, but the transmission remains incomplete.
3. The TRRn bit must be set in the exact SYSCLKOUT cycle where the CAN module is in idle state for one cycle. The CAN module is said to be in idle state when it is not in the process of receiving/transmitting data.

If these conditions occur, then the TRRn and TRSn bits for the mailbox will clear t_{clr} SYSCLKOUT cycles after the TRR bit is set where:

$$t_{clr} = ((\text{mailbox\_number})*2)+3 \text{ SYSCLKOUT cycles}$$

The TAn and AAn bits will not be set if this condition occurs. Normally, either the TA or AA bit sets after the TRR bit goes to zero.

Workaround: When this problem occurs, the TRRn and TRSn bits will clear within t_{clr} SYSCLKOUT cycles. To check for this condition, first disable the interrupts. Check the TRRn bit tclr SYSCLKOUT cycles after setting the TRRn bit to make sure it is still set. A set TRRn bit indicates that the problem did not occur.

If the TRRn bit is cleared, it could be because of the normal end of a message and the corresponding TAn or AAn bit is set. Check both the TAn and AAn bits. If either of the bits is set, then the problem did not occur. If they are both zero, then the problem did occur. Handle the condition like the interrupt service routine would except that the AAn bit does not need clearing now.

If the TAn or AAn bit is set, then the normal interrupt routine will happen when the interrupt is re-enabled.

Advisory — *SPI: Slave-Mode Operation*

Revision(s) Affected: 0, A, B

Details: When in slave mode, the SPI does not resynchronize received words based on SPISTE. A spurious SPICLK pulse could therefore throw the data stream out of sync.

Workaround: If the circuit board is not noisy enough to generate spurious SPICLK pulses, then this is not an issue. If noise is an issue, then the McBSP in SPI-slave mode may be used, since the McBSP resynchronizes on each new word.

Advisory — *WD: WDFLAG Bit Does Not Work as Intended*

Revision(s) Affected: 0, A, B

Details: The WDFLAG bit in R281x devices cannot be used to reliably distinguish a watchdog-initiated reset from a power-on (or warm) reset. This is because the device expects the $\overline{\text{XRS}}$ pin to be pulled high (by the external reset circuit) within 4 SYSCLKOUT cycles (8 OSCLK cycles at power up) after the end of the watchdog-initiated reset pulse, which is 512 OSCCLK cycles. Most of the external $\overline{\text{XRS}}$ circuits cannot provide the fast rise-time requirement (due to the capacitance); therefore, the WDFLAG bit should not be used in applications .

Workaround: None. This bit should not be used.

Advisory *McBSP: Receive FIFO Read Conflict*

Revision(s) Affected: 0, A, B

Details: The McBSP peripheral operates with or without FIFOs. The receive FIFO has interrupt generation logic that initiates interrupts based on the 5-bit FIFO status bits (12−8) and interrupt level bits (4−0) in the MFFRX register.

If the CPU reads the receive FIFO while the McBSP module writes new data into the FIFO, there is a potential conflict. The CPU read will not be stalled and read data will not be valid. The FIFO write gets the priority. The receive FIFO will be updated after every word is received in DRR2/1 registers. The DRR2/1 register update time will primarily depend on the word size and CLKR rate. For example for 8-bit word, it should be typically 8 times the CLKR cycle time. This conflict will be more pronounced if data transferred on the receive channel is back-to-back with no delays between words.

Workaround: The receive FIFOs should be read based on receive interrupts and within the next word receive time. To avoid the read conflict, additional checks could be used before initiating receive FIFO read. In most McBSP configurations, the FSR is a receiving sync pulse either active high or low (based on the FSR polarity bit) and will go inactive during word transfer time. These active and inactive phases can be detected by checking the FSR flag bit MCFFST (bit 3) register or checking the status of the FSR pin. See the FSR flag bit description for details.

Advisory *McBSP: Read Operations Decrement the McBSP FIFO*

Revision(s) Affected: 0, A, B

Details: A read operation from any of the following locations will cause the McBSP receive FIFO contents to decrement by 1, as if the McBSP DRR1 register had been read:

- 0x7001 Reserved
- 0x7401 EV−A T1CNT
- 0x7C01 Reserved

The actual value read from the location is correct and is not affected by this issue.

Workaround(s):

1. Ensure that the McBSP receive FIFO is empty before performing any read operation from any of these addresses.
2. If McBSP traffic is common in the application and a timer count needs to be monitored, consider using a timer other then EV Timer1.

Advisory — *EV: QEP Circuit*

Revision(s) Affected: 0, A, B

Details: After a DSP reset, the QEP module fails to detect the first transition that occurs on QEP input pins. (This problem also manifests itself when an external clock is used for the EV timers.) Therefore, if the first transition occurs after a GP timer has been initialized and enabled as the QEP counter (i.e., to use QEP as source of clock), the first transition will not be counted by the GP timer. The result is an error of one count in the GP timer out of a total of 1024 counts for a 256-line encoder, or 4096 counts for a 1024-line encoder. However, the issue is not a concern under any of the following conditions:

1. *The first transition happens* ***before*** *the GP timer is initialized and enabled as QEP counter.* This ensures that all transitions are counted after initialization.
2. *After the first index pulse is received and if the index pulse is used to recalibrate the GP Timer (through capture interrupt).* The recalibration corrects the error in the GP timer; therefore, from the time the first index pulse is received, the QEP counter becomes accurate.

Workaround(s):

1. Make the first transition happen before the GP timer is initialized and enabled as QEP counter. This is usually the case because typically the rotor shaft is locked to a known position before the GP timer is initialized. Locking the rotor shaft will generate transitions on QEP input pins, unless the rotor shaft is exactly aligned to the known position (which is a rare case). Disturbing the rotor shaft on purpose takes care of the rare case.
2. Use the index pulse of the encoder to recalibrate the GP timer used as QEP counter.
3. The counter has to be forced to count before the application actually uses the QEP. During initialization, configure the internal clock (HSPCLK) to be the counter source. After the first count is done, the counter should be reconfigured for external signals (QEP/TCLKIN) and reset to 0. Now the counter will also count the first edge of the QEP.

Advisory — *Clocking: Logic-High Level for XCLKIN Pin*

Revision(s) Affected: 0, A, B

Details: This advisory is applicable only when an external oscillator is used to clock the device. The X1/XCLKIN pin is referenced to the core power supply (V_{DD}), rather than the 3.3-V I/O supply (V_{DDIO}). Therefore, the logic-high level for the input clock should not exceed V_{DD}. This requirement remains the same for future silicon revisions as well.

Workaround: A clamping diode may be used to clamp a buffered clock signal to ensure that the logic-high level does not exceed V_{DD} (1.8 V or 1.9 V). Otherwise, 1.8-V oscillators may be used.

Advisory *ADC: Reserved Bits in Autosequence Status Register (ADCASEQSR)*

Revision(s) Affected: 0, A, B

Details: SEQ2 STATE2−0 and SEQ1 STATE3−0 bit fields (bits 6 through 0) are the pointers of SEQ2 and SEQ1, respectively. These bits are reserved for TI testing and should not be used in customer applications.

Workaround: None

Advisory *ADC: Result Register Update Delay*

Revision(s) Affected: 0, A, B

Details: The ADC result status flags INT_SEQ1 and INT_SEQ2 bit fields (bits 0 and 1 respectively) in the ADC_ST_FLG register indicate the availability of new ADC results after conversions and initiation of the ADC interrupts.

The update of the ADC result register requires one extra ADC cycle to complete after the status flags INT_SEQ1 and INT_SEQ2 bit(s) are set. The result of reading the result register prior to this extra cycle will result in old data being read (reset value/previous conversion result).

If auto-sequencers are enabled with a non-zero value in the MAXCONV register, the last result register update takes an additional ADC cycle from the time the INT_SEQ1 or INT_SEQ2 flag is set.

Workaround: Delay the read of the ADC result register(s) by at least one ADC clock period. This delay can be implemented by using software delay loops.

If the ADC result register(s) are read using the ADC interrupt, rather than polling, the wait period introduced by the ISR (interrupt service routine) could minimize the delay needed in software. This ISR branching delay is generally greater than 8 SYSCLKOUT cycles.

The ratio of the ADC clock (ADCCLK) to the CPU clock (SYSCLKOUT) determines the size of the software delay. For example, if ADCCLK = 10 MHz the software delay should be at least 100 ns.

Timing example to estimate the software delay:

1) Get the HSPCLK prescaler value − HISPCP
2) Get the ADCCLK prescaler value − ADCCLKPS
3) Get the CPS (ADCCTRL1[7]) value − CPS
4) Software wait-period in CPU cycles (SYSCLKOUT) before the ADC result register read is defined as:

```
Software wait  = (HISPCP *2) * (ADCCLKPS * 2) * (CPS +1)  cycles

If HISPCP  or ADCCLKPS is 0, then the respective terms should be (HISPCP +1)
or (ADCCLKPS+1)
```

Advisory — *ADC: Sequencer Reset While Dual Sequencers Are Running*

Revision(s) Affected: 0, A, B

Details: In the TMS320R2811/TMS320R2812 on-chip ADC, there are two sequencers for performing ADC conversions; SEQ1 and SEQ2. If one of the sequencers is reset while the other sequencer is running, it will result in the running sequencer never completing its current sequence. The sequencer busy bit (bit 3/bit2 in ADCST register) for the sequencer will remain active and an “End−of−sequence (EOS)” interrupt for the running sequencer will never be generated. For example, if SEQ1 is reset while SEQ2 is performing a sequence, then SEQ2 will never complete.

Workaround: If dual sequencers are enabled, then the software handling the ADC module should make sure that SEQ1 BSY/ SEQ2 BSY bits are not set before performing a reset of either sequencer.

Advisory — *ADC: EOS BUF2 and EOS BUF1 Bits in ADCST Register Corrupted When INT MOD SEQ1 and INT MOD SEQ2 Are Used Simultaneously*

Revision(s) Affected: 0, A, B

Details: Setting the INT MOD SEQx bit in ADCTRL2 causes the ADC wrapper to generate INT SEQx at the end of every other conversion rather than at the end of every conversion. Also EOS BUFx will be set at the end of every conversion to track the status of the SEQx in use. However, if both INT MOD SEQ1 and INT MOD SEQ2 are set at the same time, then a conversion on SEQ1 will set both EOS BUF1 and EOS BUF2. This will cause INT SEQ2 to be generated after the next SEQ2 completion. The same applies if a conversion on SEQ2 happens first.

Workaround: Do not set INT MOD SEQ1 and INT MOD SEQ2 at the same time. Independently, if either INT MOD SEQ1 or INT MOD SEQ2 is set, this feature works as expected.

3 Documentation Support

For device-specific data sheets and related documentation, visit the TI web site at: **http://www.ti.com**

For further information regarding the TMS320R2811 and TMS320R2812, please see the following publication:

- *TMS320R2811 and TMS320R2812 Digital Signal Processors* data manual (literature number SPRS257)

4 Revision History

This revision history highlights the technical changes made to the SPRZ226B to make it an SPRZ226C revision.

Scope: Added silicon revisions A and B.
Added two new advisories.

PAGE(S) NO.	ADDITIONS/CHANGES/DELETIONS
Global	Added silicon revisions A and B.
10	Added “SCI: Bootloader Does Not Clear the ABD Bit After Auto-Baud Lock” advisory
13	Added “eCAN: Abort Acknowledge Bit Not Set” advisory

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	Broadband	www.ti.com/broadband
DSP	dsp.ti.com	Digital Control	www.ti.com/digitalcontrol
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Military	www.ti.com/military
Logic	logic.ti.com	Optical Networking	www.ti.com/opticalnetwork
Power Mgmt	power.ti.com	Security	www.ti.com/security
Microcontrollers	microcontroller.ti.com	Telephony	www.ti.com/telephony
RFID	www.ti-rfid.com	Video & Imaging	www.ti.com/video
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Wireless	www.ti.com/wireless

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