TMS320F280x, TMS320C280x, and TMS320F2801x DSC

Silicon Errata

Literature Number: SPRZ171J
December 2004 – Revised February 2008

TEXAS INSTRUMENTS

Contents

List of Figures

List of Tables

280x/2801x Silicon Errata

1 Introduction

This document describes the silicon updates to the functional specifications for the TMS320C2801, TMS320C2802, TMS320F2801, TMS320F2802, TMS320F2806, TMS320F2808, TMS320F2809, TMS320F28015, and TMS320F28016 digital signal processors (DSPs). The TMS320F2809 device is new in this silicon errata.

The updates are applicable to:

- 100-ball MicroStar BGA™, GGM, and ZGM suffix
- 100-pin thin quad flatpack, PZ suffix

Throughout this document, the device names are abbreviated as follows:

- F280x or TMS320F280x refers to TMS320F2809, TMS320F2808, TMS320F2806, TMS320F2802 and TMS320F2801 silicon
- C280x or TMS320C280x refers to TMS320C2802 & TMS320C2801 silicon.
- F2801x or TMS320F2801x refers to TMS320F28015 & TMS320F28016 silicon.

Throughout this document, any reference to F2801 and F2802 devices includes both 60-MHz and 100-MHz versions.

2 Device and Development Tool Support Nomenclature

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all [TMS320] DSP devices and support tools. Each TMS320™ DSP commercial family member has one of three prefixes: TMX, TMP, or TMS (e.g., **TMS**320F2808). Texas Instruments recommends two of three possible prefix designators for its support tools: TMDX and TMDS. These prefixes represent evolutionary stages of product development from engineering prototypes (TMX/TMDX) through fully qualified production devices/tools (TMS/TMDS).

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

TMP Final silicon die that conforms to the device's electrical specifications but has not completed quality and reliability verification

TMS Fully qualified production device

Support tool development evolutionary flow:

TMDX Development-support product that has not yet completed Texas Instruments internal qualification testing

TMDS Fully qualified development-support product

TMX and TMP devices and TMDX development-support tools are shipped against the following disclaimer:
"Developmental product is intended for internal evaluation purposes."

TMS devices and TMDS development-support tools have been characterized fully, and the quality and reliability of the device have been demonstrated fully. TI's standard warranty applies.

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

TI device nomenclature also includes a suffix with the device family name. This suffix indicates the package type (for example, PBK) and temperature range (for example, A).

3 Device Markings

Figure 1 provides an example of the TMS320F280x device markings and defines each of the markings. The device revision can be determined by the symbols marked on the top of the package as shown in Figure 1. Some prototype devices may have markings different from those illustrated. Figure 2 shows an example of device nomenclature.

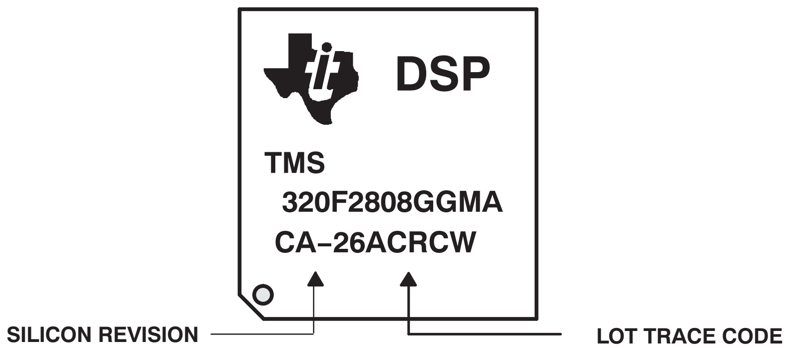


Figure 1. Example of Device Markings

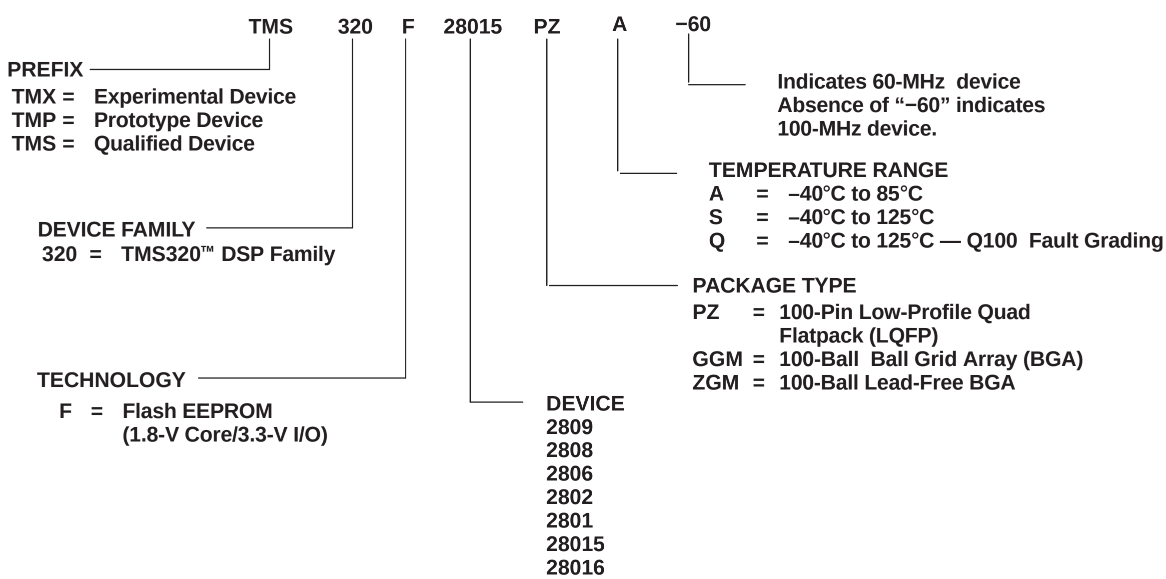


Figure 2. Example of Device Nomenclature

Table 1. Determining Silicon Revision From Lot Trace Code (F2809)

SECOND LETTER IN PREFIX OF LOT TRACE CODE	SILICON REVISION	REVISION ID (0x0883)	F2809 COMMENTS
Blank (no second letter in prefix)	Indicates Revision 0	0x0000	This silicon revision is available as TMS

Table 2. Determining Silicon Revision From Lot Trace Code (C2801 and C2802)

SECOND LETTER IN PREFIX OF LOT TRACE CODE	SILICON REVISION	REVISION ID (0x0883)	C2801, C2802 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

Table 3. Determining Silicon Revision From Lot Trace Code (F2801, F2802, F2806, F2808, F28015 and F28016)

SECOND LETTER IN PREFIX OF LOT TRACE CODE	SILICON REVISION	REVISION ID (0x0883)	F2801, F2802, F2806, and F2808 COMMENTS	F28015 and F28016 COMMENTS
Blank (no second letter in prefix)	Indicates Revision 0	0x0000	This silicon revision is available as TMX only.	TI internal only
A	Indicates Revision A	0x0001	This silicon revision is available as TMX only.	TI internal only
B	Indicates Revision B	0x0002	This silicon revision is available as TMS.	TI internal only
C	Indicates Revision C	0x0003	This silicon revision is available as TMS.	This silicon revision is available as TMS.

4 Silicon Change Overview

Table 6 and Table 5 list the change(s) made to each silicon revision.

Table 4. TMS320F2809 Silicon Change Overview

REVISION	CHANGES MADE
0	First silicon release. (This is functionally equivalent to Revision C of the TMS320F280x silicon.)

Table 5. TMS320C2802 and TMS320C2801 Silicon Change Overview

REVISION	CHANGES MADE
A	TMS silicon (This is functionally equivalent to Revision C of the TMS320F280x silicon.)
0	First silicon release. (This is functionally equivalent to Revision B of the TMS320F280x silicon - TI Internal only)

Table 6. TMS320F2808, TMS320F2806, TMS320F2802, TMS320F2801, and TMS320F2801x Silicon Change Overview

REVISION	CHANGES MADE
C	The following advisories were fixed: • Watchdog module limitation • ADC crosstalk issue
B	• First TMS silicon release. • Flash tools: All flash tools must be updated to use F280x flash API v3.00 (SPRC193) or later. This API is backward-compatible with all previous silicon versions. Previous API versions will no longer work. • The default state of the internal pullup resistors for pins GPIO0 to GPIO11 changed from enabled to disabled. These pins correspond to ePWM output pins. The default state of the internal pullup resistors for pins GPIO12 to GPIO34 remains as enabled. • The following advisory was fixed: – GPIO pin behavior at power-up
A	The following advisories were fixed: • Boot ROM – configuration of pins as asynchronous • eCAN – boot mode in boot ROM • ADC – Initial Conversion Latency
0	First silicon release.

5 Known Design Marginality/Exceptions to Functional Specifications

The table of contents for advisories is shown in Table 7.

Table 7. Advisory List

Advisory — *Input Clock: Device startup using XCLKIN input*

Revision(s) Affected 0, applicable only to F2809 silicon

Details When clock to the device is supplied using the XCLKIN pin, device may intermittently fail to startup correctly.

Workaround(s) Do not use the XCLKIN pin to supply clock to the device. Instead, use either a crystal/resonator or a 1.8-V external oscillator on the X1 pin to clock the device. This will be fixed in the next revision of the silicon.

Advisory — *Memory: Flash and OTP Prefetch Buffer Overflow*

Revision(s) Affected

0 on F2809 silicon

0, A on C280x silicon

0, A, B, C on F2801, F2802, F2806, F2808, and F2801x silicon

Details

This advisory applies to code executing from flash or OTP with the flash prefetch buffer enabled. On ROM devices this applies to the ROM that replaces flash and OTP.

The flash prefetch buffer may overflow if a SBF or BF instruction is within eight 16-bit words preceding an operation using indirect or direct program-memory addressing. The window for which this can occur is shown below:

```
Address
Offset

0x0000    BF LSW (32-bit opcode)
0x0001    BF MSW or SBF (16-bit opcode)
-------------------------------------
0x0002    SBF/BF + 1 word         ↑
0x0003    SBF/BF + 2 words
0x0004    SBF/BF + 3 words     If an instruction within this window
0x0005    SBF/BF + 4 words     uses program-memory addressing, it
0x0006    SBF/BF + 5 words     can cause the flash prefetch buffer to
0x0007    SBF/BF + 6 words     overflow.
0x0008    SBF/BF + 7 words
0x0009    SBF/BF + 8 words        ↓
------------------------------------
0x0010    SBF/BF + 9 words
```

Whether or not an overflow actually occurs depends on the instruction sequence, flash wait states and CPU pipeline stalls. If an overflow occurs it will result in execution of invalid opcodes. Instructions that use program-memory addressing include MAC/XMAC, DMAC/XMACD, QMACL, IMACL, PREAD/XPREAD and PWRITE/XPWRITE.

Workaround(s)

1. Hand-coded assembly:

Use the SB/B instructions instead of SBF/BF for code targeted to execute from flash or OTP. The SB/B instructions are more efficient in wait-stated memory so a performance improvement may also be seen.

2. Compiler-generated assembly:

Use the compiler switch -me to force the compiler to generate SB/B instructions instead of SBF/BF instructions. In heavily wait stated memory the SB/B instructions are more efficient than SBF/BF. In SARAM the SBF/BF instructions are more efficient. Therefore, this switch should be applied as follows:

- Use the compiler switch -me on source code that runs from flash or OTP.
- Do not use the compiler switch -me on source code that runs from SARAM.
- Use -me If a file contains functions that runs from flash as well as functions that run from SARAM.

The -me switch is available in C28x compiler as of V4.1.4 and V5.0 beta3.

Advisory — *Memory: Prefetching Beyond Valid Memory*

Revision(s) Affected

0 on F2809 silicon

0, A on C280x silicon

0, A, B, C on F2801, F2802, F2806, F2808, and F2801x silicon

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 and all memory types (Flash/ROM, OTP, 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 — *ADC: Simultaneous Sampling Latency*

Revision(s) Affected

0 on F2809 silicon

0, A on C280x silicon

0, A, B, C on , F2801, F2802, F2806, F2808, and F2801x silicon

Details

When the ADC conversions are initiated in simultaneous mode, the first sample pair will not give correct conversion results.

Workaround(s)

1. If the ADC is used with a sampling window ≤ 160 nS, then the first sample pair must be discarded and a second sample of the same pair must be taken. For instance, if the sequencer is set to sample channel A0:B0/A1:B1/A2:B2 in that order, then load the sequencer with A0:B0/A0:B0/A1:B1/A2:B2 and only use the last three conversions.

2. If the ADC is used with a sampling window greater than 160 ns, there is no issue.

Advisory — *ADC: Initial Conversion Latency*

Revision(s) Affected

0 on F2808, F2806, F2801

Details

When the ADC conversions are initiated by any source of trigger, the first two samples may not be correct conversion results.

Workaround(s)

1. If the ADC is set to convert at 1 mega sample per second (MSPS) or higher, discard the first two samples

   For instance, if the sequencer is set to sample channel A0/A1/A2 in that order, then load the sequencer with A0/A0/A0/A1/A2 and only use the last three conversions.
2. If the ADC is set at a conversion rates below 1 MSPS, the conversion latency will give the ADC appropriate time to settle and the first conversion should be valid. Each application should validate this as acceptable in their application.

   This has been fixed in the B revision of the silicon.

Advisory — *ADC: ADC A Channel to B Channel Crosstalk in Simultaneous Mode*

Revision(s) Affected

0 on TMS320C280x silicon

0, A, B on F2801, F2802, F2806, F2808, and F2801x silicon

Details

When the ADC is used in simultaneous mode, voltage present on an A channel will impact the conversion value of the associated B channel. The A channel is unaffected by the B channel.

For example, if A4/B4 are being sampled simultaneously, the converted value of B4 will have a dc error associated with the value present on A4. Voltages on the other A channels have no impact on B4; likewise, A0 affects only B0, A1 affects only B1, etc.

The effect of An on Bn is deterministic; from 0 to 16 codes of artificial dc increase. For example, if A channel is at 0 V, the converted B channel value will be unaffected. If An is at 1.5 V, then the Bn converted value will read 8 counts too high.

Workaround(s)

Due to the deterministic nature of the coupling from An to Bn, a simple subtraction can be made from the B channel based on the A channel result.

Formula given as:

$$BnC = BnM - (An / 256)$$

BnC = Corrected result for Bn channel

BnM = Measured result for Bn channel.

Since the effect of A on B is a pure dc adder, there is no impact to linearity of the B channel. Gain and offset errors are only nominally impacted, ± 2 LSBs.

Revision C silicon has a design change to address this errata. The crosstalk will be within the datasheet specification of channel-to-channel offset. See the most recent version of the *TMS320F2809, TMS320F2808, TMS320F2806, TMS320F2802, TMS320F2801, UCD9501, TMS320C2802, TMS320C2801, and TMS320F2801x DSPs Data Manual* (literature number SPRS230) for more information.

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

Revision(s) Affected

0 on F2809 silicon

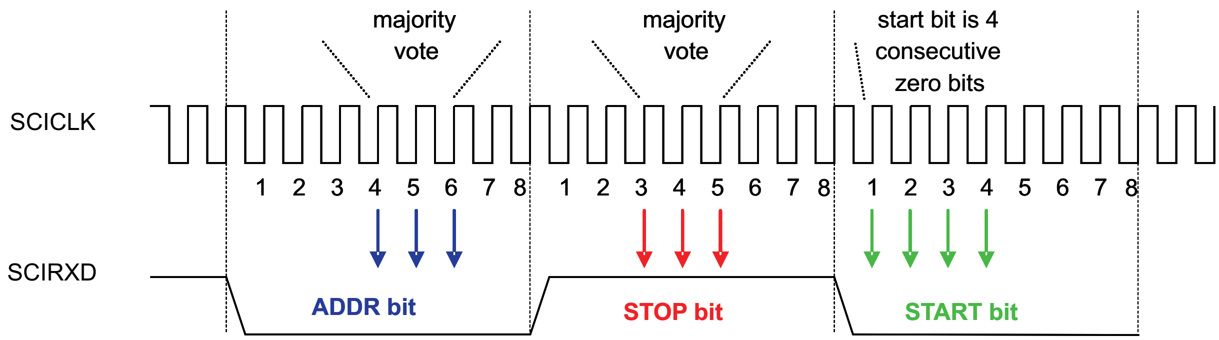
0, A on C280x silicon

0, A, B, C on F2801, F2802, F2806, F2808, and F2801x silicon

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 the false START bit to occur since the 4th sub-sample for the start bit may be sensed low.

Expected Operation:



Erroneous Operation:

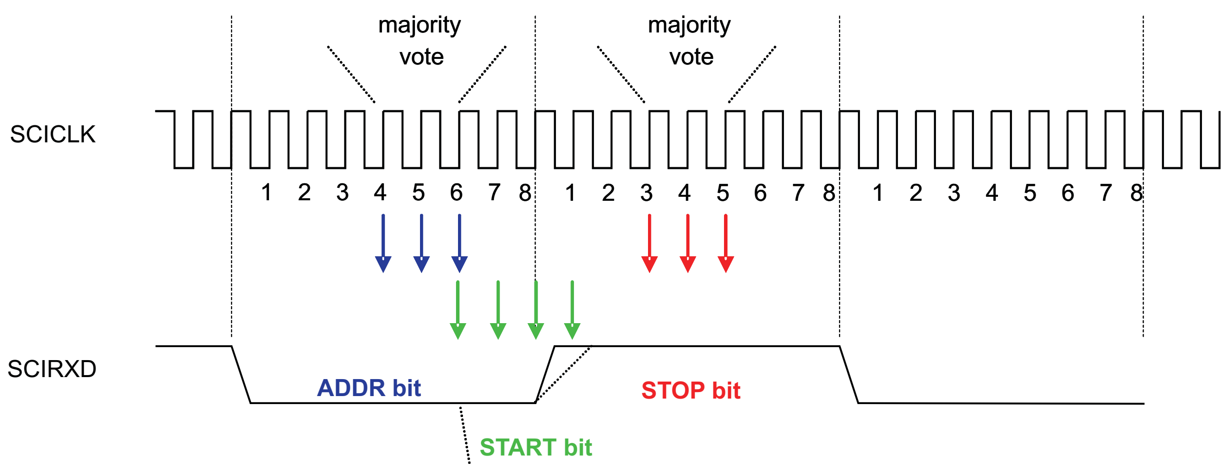


Figure 3. Difference Between Expected and Erroneous Operation of START Bit

Workaround(s)

Program the baud rate of the SCI to be slightly slower than the actual. This will cause the 4th 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 on F2809 silicon

0, A on C280x silicon

0, A, B, C on F2801, F2802, F2806, F2808, and F2801x silicon

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: When the CAN option is invoked in the boot ROM, the code may hang occasionally*

Revision(s) Affected

0 on F2809 silicon

0, A on C2801 silicon

0, A, B, C on C280x, F2801, F2802, F2806, F2808, and F2801x silicon

Details

This happens because of a 16-bit R/W employed to check the status of the CCE bit in the boot-ROM code. Since 16-bit R/W returns undefined values, the code may get stuck in a loop, mistakenly reading the value of the bit to be opposite of what it really is.

Workaround(s)

A power-cycling could fix this issue; however, since this is a random phenomenon, it may not work consistently. An option would be to burn the CAN boot-load code in OTP.

Advisory — *eCAN: eCAN-A Boot Mode in Boot ROM*

Revision(s) Affected

0, A on F2808, F2806, F2801

Details

The eCAN-A boot mode in boot ROM does not work as intended. This is because the IDE and AME bits of the MSGID1 register are not initialized by the boot loader code. Since these bits can come up as 0 or a 1, frames transmitted by the host may not be received on the 2808.

Workaround(s)

This has been fixed in the B revision of the silicon. If the existing bootloader is to be used for developing an application, be certain that the IDE and AME bits are 0 before proceeding to use the eCAN-A mode of the bootloader to be sure that a standard identifier frame with an ID of 1 is received by the eCAN-A module.

Advisory — *eCAN: Abort Acknowledge Bit Not Set*

Revision(s) Affected

0 on F2809 silicon

0, A on C280x silicon

0, A, B, C on F2801, F2802, F2806, F2808, and F2801x silicon

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(s)

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 t_{clr} 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 one 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 — *WD: Change to Watchdog Module: Bad Key Writes to WDKEY No Longer Cause RESET/Interrupt to be Generated*

Revision(s) Affected

0 on F2809 silicon

A on C280x silicon

C on F2801, F2802, F2806, F2808, and F2801x silicon

Details

The “Bad Key Detect” function of the WDKEY register has been disabled. When using the Watchdog (WD) module, a write of anything other than 0x55 or 0xAA to the WDKEY register will have no effect. See the *TMS320x280x, 2801x, 2804x DSP System Control and Interrupts Reference Guide* (literature number SPRU712) for more information.

Workaround(s)

To trigger an immediate reset or interrupt, perform an invalid write to the WDCHK bits in the WDCR register.

Advisory — *WD: Limitation on Watchdog Module: Corrupted Watchdog Key Writes*

Revision(s) Affected

0 on C280x silicon

0, A, B on F2801, F2802, F2806, F2808, and F2801x silicon

Details

When using the on-chip PLL (PLLCR ≠ 0), writes of the 0x55/0xAA sequence to WDKEY register may be corrupted. Although the watchdog counter will be reset correctly, this will cause a Watchdog (WD) interrupt or reset depending on the state of the WDENINT bit in the SCSR register.

Workaround(s)

1. Use PLL in bypass mode (PLLCR = 0) or PLL off mode (PLLOFF = 1 in PLLSTS register). In this case, CLKINDIV in the PLLSTS register can be set or cleared. This is valid for both the WD interrupt and the WD reset cases.
2. **Case 1: Applications Using the WD Interrupt**

   Implement a software function (ServiceWatchDog) that performs the writes of 0x55 and 0xAA to the WDKEY register, as shown below. The WD interrupt (WAKEINT in the PIE) is remapped to a pseudo interrupt service routine (ISR). The ServiceWatchDog routine will deterministically force a WD interrupt each time the function is called. This forced interrupt will be serviced by the pseudo ISR. The pseudo ISR will then acknowledge the interrupt and remap the WAKEINT interrupt back to the normal WD ISR.

   Note: The WDINT signal, once triggered, will stay active low for 512 OSC Clock cycles. If another WD event (timeout or bad key write) comes before this signal has gone inactive high, the event will not be captured by the WD module. See the *TMS320x280x, 2801x, 2804x DSP System Control and Interrupts Reference Guide* (literature number SPRU712) section on Watchdog Reset or Watchdog Interrupt Mode for more information

   Case 2: Applications Using the WD Reset

   This case uses the interrupt feature of the WD module to work around the possible corruption of the WDKEY register and to service any WD events that would normally trigger a reset. Applications that only used the reset feature of the WD will now need to properly map and enable the WAKEINT interrupt in the PIE. Applications will also need to enable the interrupt function of the WD by setting the WDENINT bit in the SCSR register. The reset feature of the WD will only be enabled inside the WatchdogInterrupt interrupt service routine (ISR) and triggered when a true WD event occurs, either from a WD timeout or an incorrect write to the WDKEY register or the WDCHK bits in the WDCR register. Inside the ISR an incorrect value is written to the WDKEY to force the WD reset. Since the WD reset is gated by servicing the WD interrupt, applications must re-enable WD interrupts via the PIEIER and the INTM bit in ST1 inside other ISRs. In order to service the WD(reset the WD counter) during normal operation, implement a software function (ServiceWatchDog) that performs the writes of 0x55 and 0xAA to the WDKEY register, as shown below. The WD interrupt (WAKEINT in the PIE) is remapped to a pseudo ISR. The ServiceWatchDog routine will deterministically force a WD interrupt each time the function is called. This forced interrupt will be serviced by the pseudo ISR. The pseudo ISR will then acknowledge the interrupt and remap the WAKEINT interrupt back to the normal WD ISR.

Code example for Case 1

```
void ServiceWatchdog (void)
{
EALLOW;
DINT;                        // Disable Global Interrupts
if(SysCtrlRegs.WDCNTR < 254) // If watchdog counter is
                             // less then 254, then there
                             // is enough time to use the
                             // service watchdog function;
                             // otherwise, assume it is
                             // too late and let the
                             // watchdog time out.
{
PieVectTable.WAKEINT = &PseudoWatchdogInterrupt; // Remap
                             // vector to pseudo routine
SysCtrlRegs.WDKEY = 0x0000;  // Force an interrupt always
SysCtrlRegs.WDKEY = 0x0055;
SysCtrlRegs.WDKEY = 0x00AA;  // This will clear the
                             // watchdog counter

}
EINT;                        // Enable global interrupts
EDIS;
}
interrupt void PseudoWatchdogInterrupt(void)
{
EALLOW;
PieVectTable.WAKEINT = &WatchdogInterrupt; // This will clear
                             // PIEIFR.INT1.8 flag
                             // and remap back to
                             // proper service
                             // routine
PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;
EDIS;
}
interrupt void WatchdogInterrupt(void)
{
                            // Proper Watchdog Interrupt;
}
```

Code example for Case 2

```
void ServiceWatchdog (void)
{
EALLOW;
DINT;                         // Disable Global Interrupts
if(SysCtrlRegs.WDCNTR < 254) // If watchdog counter is
                              // less then 254, then there
                              // is enough time to use the
                              // service watchdog function;
                              // otherwise, assume it is
                              // too late and let the
                              // watchdog time out.
{
PieVectTable.WAKEINT = &PseudoWatchdogInterrupt; // Remap
                              // vector to pseudo routine
SysCtrlRegs.WDKEY = 0x0000;  // Force an interrupt always
SysCtrlRegs.WDKEY = 0x0055;
SysCtrlRegs.WDKEY = 0x00AA;  // This will clear the
                              // watchdog counter

}
EINT;                         // Enable global interrupts
EDIS;
}
interrupt void PseudoWatchdogInterrupt(void)
{
EALLOW;
PieVectTable.WAKEINT = &WatchdogInterrupt; // This will clear
                             // PIEIFR.INT1.8 flag
                             // and remap back to
                             // proper service
                             // routine
PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;
EDIS;
}
interrupt void WatchdogInterrupt(void)
{
EALLOW;
SysCtrlRegs.SCSR = 0x0000;  // Set the WD to generate WDRSTn
SysCtrlRegs.WDKEY = 0x0000; // In case WDINTn is not low, force
                             // the reset with a bad key write
EDIS;
                             // Proper Watchdog Interrupt;
}
```

Advisory *GPIO: Pin Behavior at Power-up*

Revision(s) Affected 0, A on F2808, F2806, F2801

Details GPIO0-13, GPIO20-GPIO21, and GPIO25-31 can potentially drive a signal out while the device VDD and VDDIO pins are powering up, prior to the DSP receiving the first valid input clock from the X1 or XCLKIN pin. Once VDD and VDDIO are fully powered and the first clock pulse is received, the device will place these pins into a high impedance state.

Workaround(s) None. The synchronous nature of these pins has been removed in the B revision of the silicon.

Advisory *Boot ROM: Configuration Change in Boot ROM*

Revision(s) Affected 0, A on F2808, F2806, F2801

Details In the input configuration, all GPIO pins come up synchronized to SYSCLKOUT. This is different compared to the TMS320x281x

Workaround(s) This has been fixed in the B revision of the silicon. The boot ROM will configure the peripheral pins used for asynchronous mode operation.

6 Documentation Support

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

To access documentation on the web site:

1. Go to http://www.ti.com
2. Click on *DSP Product Tree*
3. Click on the *C2000 platform*
4. Click on *C28x* DSPs
5. Click on a device name and then click on the documentation type you prefer.

For further information regarding the 280x DSPs, please see the *TMS320F2809, TMS320F2808, TMS320F2806, TMS320F2802, TMS320F2801, TMS320C2802, TMS320C2801, and TMS320F2801x DSPs Data Manual* (literature number SPRS230).

Appendix A Revision History

This silicon errata was revised from SPRZ171J to SPRZ171I. Table A-1 lists the technical changes made in this revision.

Table A-1. Changes Made in This Revision

Location	Additions, Deletions, Modifications
Input Clock	Added advisory on device startup using XCLKIN input

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

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