Errata

TMS320F280x, TMS320C280x, TMS320F2801x DSPs Silicon revisions C, B, A, 0

1 Introduction

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

The updates are applicable to:

- 100-ball MicroStar BGA™, GGM, ZGM, GBA, and NMF 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 and TMS320C2801 silicon.
- F2801x or TMS320F2801x refers to TMS320F28016 and TMS320F28015 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 (for example, **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, GBA) and temperature range (for example, A).

3 Device Markings

Figure 3-1 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 3-1. Some prototype devices may have markings different from those illustrated. Figure 3-2 shows an example of device nomenclature.

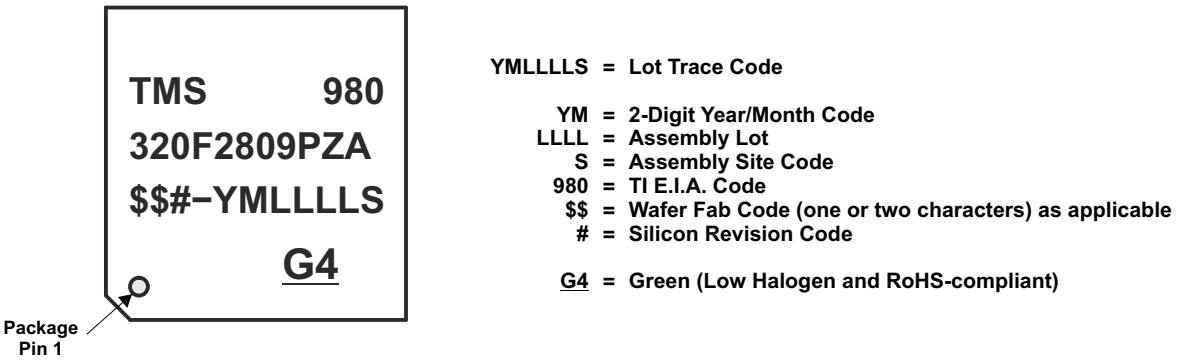


Figure 3-1. Example of Device Markings

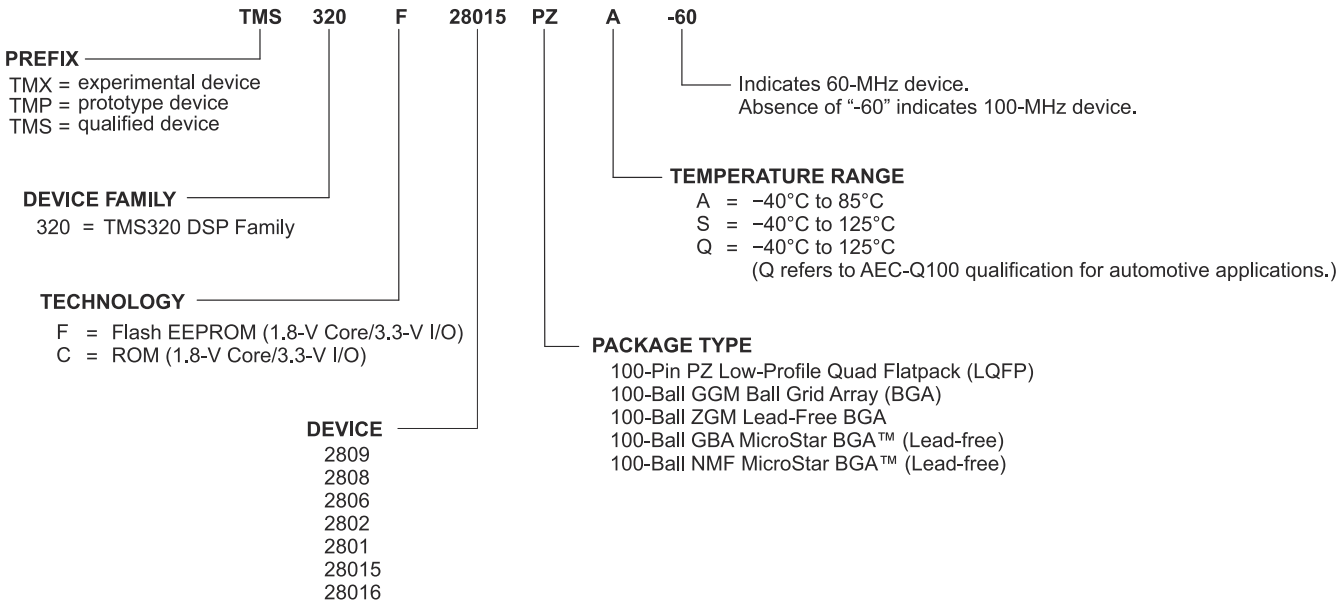


Figure 3-2. Example of Device Nomenclature

Table 3-1. Determining Silicon Revision From Lot Trace Code (F2809)

SILICON REVISION CODE	SILICON REVISION	REVISION ID Address: 0x0883	F2809 COMMENTS
Blank (no second letter in prefix)	Indicates Revision 0	0x0000	This silicon revision is available as TMS.
A	Indicates Revision A	0x0001	This silicon revision is available as TMS.

Table 3-2. Determining Silicon Revision From Lot Trace Code (C2801 and C2802)

SILICON REVISION CODE	SILICON REVISION	REVISION ID Address: 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 3-3. Determining Silicon Revision From Lot Trace Code (F2801, F2802, F2806, F2808, F28015 and F28016)

SILICON REVISION CODE	SILICON REVISION	REVISION ID Address: 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 4-1 to Table 4-3 list the change(s) made to each silicon revision.

Table 4-1. TMS320F2809 Silicon Change Overview

REVISION	CHANGES MADE
A	The following advisory was fixed: "Input Clock: Device Startup Using XCLKIN Input".
0	First silicon release. (This is functionally equivalent to Revision C of the TMS320F280x silicon.)

Table 4-2. 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 4-3. 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 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 Usage Notes and Known Design Exceptions to Functional Specifications

5.1 Usage Notes

Usage notes highlight and describe particular situations where the device's behavior may not match presumed or documented behavior. This may include behaviors that affect device performance or functional correctness. These usage notes will be incorporated into future documentation updates for the device (such as the device-specific data sheet), and the behaviors they describe will not be altered in future silicon revisions.

Table 5-1 through Table 5-3 show which silicon revision(s) are affected by each usage note.

Table 5-1. List of Usage Notes for F2809, C2801, and C2802

TITLE	SILICON REVISION(S) AFFECTED(1)					
	F2809		C2801		C2802	
	0	A	0	A	0	A
PIE: Spurious Nested Interrupt After Back-to-Back PIEACK Write and Manual CPU Interrupt Mask Clear	Y	Y	Y	Y	Y	Y

(1) Y = Yes

Table 5-2. List of Usage Notes for F2801, F2802, F2806, F2808

TITLE	SILICON REVISION(S) AFFECTED(1)															
	F2801				F2802				F2806				F2808			
	0	A	B	C	0	A	B	C	0	A	B	C	0	A	B	C
PIE: Spurious Nested Interrupt After Back-to-Back PIEACK Write and Manual CPU Interrupt Mask Clear	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

(1) Y = Yes

Table 5-3. List of Usage Notes for F28015 and F28016

TITLE	SILICON REVISION(S) AFFECTED(1)							
	F28015				F28016			
	0	A	B	C	0	A	B	C
PIE: Spurious Nested Interrupt After Back-to-Back PIEACK Write and Manual CPU Interrupt Mask Clear	Y	Y	Y	Y	Y	Y	Y	Y

(1) Y = Yes

5.1.1 PIE: Spurious Nested Interrupt After Back-to-Back PIEACK Write and Manual CPU Interrupt Mask Clear Usage Note

Revision(s) Affected: 0, A on F2809 silicon
0, A on C280x silicon
0, A, B, C on F2801, F2802, F2806, F2808, and F2801x silicon

Certain code sequences used for nested interrupts allow the CPU and PIE to enter an inconsistent state that can trigger an unwanted interrupt. The conditions required to enter this state are:

1. A PIEACK clear is followed immediately by a global interrupt enable (EINT or asm(" CLRC INTM")).
2. A nested interrupt clears one or more PIEIER bits for its group.

Whether the unwanted interrupt is triggered depends on the configuration and timing of the other interrupts in the system. This is expected to be a rare or nonexistent event in most applications. If it happens, the unwanted interrupt will be the first one in the nested interrupt's PIE group, and will be triggered after the nested interrupt re-enables CPU interrupts (EINT or asm(" CLRC INTM")).

Workaround: Add a NOP between the PIEACK write and the CPU interrupt enable. Example code is shown below.

```
//Bad interrupt nesting code
PieCtrlRegs.PIEACK.all = 0xFFFF;      //Enable nesting in the PIE
EINT;                                 //Enable nesting in the CPU

//Good interrupt nesting code
PieCtrlRegs.PIEACK.all = 0xFFFF;      //Enable nesting in the PIE
asm(" NOP");                          //Wait for PIEACK to exit the pipeline
EINT;                                 //Enable nesting in the CPU
```

5.2 Known Design Exceptions to Functional Specifications

Table 5-4 through Table 5-5 show which silicon revision(s) are affected by each advisory.

Table 5-4. List of Advisories for F2809, C2801, and C2802

TITLE	SILICON REVISION(S) AFFECTED(1)					
	F2809		C2801		C2802	
	0	A	0	A	0	A
Memory: Flash and OTP Prefetch Buffer Overflow	Y	Y	Y	Y	Y	Y
Memory: Prefetching Beyond Valid Memory	Y	Y	Y	Y	Y	Y
ADC: Simultaneous Sampling Latency	Y	Y	Y	Y	Y	Y
ADC: ADC Inaccuracy at Low Frequencies	Y	Y	Y	Y	Y	Y
ADC: Initial Conversion Latency	N/A	N/A	N/A	N/A	N/A	N/A
ADC: ADC A Channel to B Channel Crosstalk in Simultaneous Mode	N/A	N/A	Y	N/A	Y	N/A
SCI: Incorrect Operation of SCI in Address Bit Mode	Y	Y	Y	Y	Y	Y
SCI: Bootloader Does Not Clear the ABD Bit After Auto-Baud Lock	Y	Y	Y	Y	Y	Y
eCAN: When the CAN Option is Invoked in the Boot ROM, the Code may Hang Occasionally	Y	Y	Y	Y	Y	Y
eCAN: eCAN-A Boot Mode in Boot ROM	N/A	N/A	N/A	N/A	N/A	N/A
eCAN: Unexpected Cessation of Transmit Operation	Y	Y	Y	Y	Y	Y
eCAN: Abort Acknowledge Bit Not Set	Y	Y	Y	Y	Y	Y
WD: Change to Watchdog Module: Bad Key Writes to WDKEY No Longer Cause RESET/Interrupt to be Generated	Y	Y	N/A	Y	N/A	Y
WD: Limitation on Watchdog Module: Corrupted Watchdog Key Writes	N/A	N/A	Y	N/A	Y	N/A
GPIO: Pin Behavior at Power-up	N/A	N/A	N/A	N/A	N/A	N/A
GPIO: GPIO Qualification	Y	Y	Y	Y	Y	Y
Boot ROM: Configuration Change in Boot ROM	N/A	N/A	N/A	N/A	N/A	N/A
Pulldown Resistor for $\overline{\text{TRST}}$ Pin	Y	Y	Y	Y	Y	Y
Input Clock: Device Startup Using XCLKIN Input	Y	N/A	N/A	N/A	N/A	N/A
eQEP: Missed First Index Event	Y	Y	Y	Y	Y	Y
eQEP: eQEP Inputs in GPIO Asynchronous Mode	Y	Y	Y	Y	Y	Y
eQEP: Position Counter Incorrectly Reset on Direction Change During Index	Y	Y	Y	Y	Y	Y

(1) Y = Yes; N/A = Not Applicable

Table 5-5. List of Advisories for F2801, F2802, F2806, F2808

TITLE	SILICON REVISION(S) AFFECTED(1)															
	F2801				F2802				F2806				F2808			
	0	A	B	C	0	A	B	C	0	A	B	C	0	A	B	C
Memory: Flash and OTP Prefetch Buffer Overflow	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Memory: Prefetching Beyond Valid Memory	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ADC: Simultaneous Sampling Latency	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ADC: ADC Inaccuracy at Low Frequencies	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
ADC: Initial Conversion Latency	Y	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Y	N/A	N/A	N/A	Y	N/A	N/A	N/A
ADC: ADC A Channel to B Channel Crosstalk in Simultaneous Mode	Y	Y	Y	N/A	Y	Y	Y	N/A	Y	Y	Y	N/A	Y	Y	Y	N/A
SCI: Incorrect Operation of SCI in Address Bit Mode	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
SCI: Bootloader Does Not Clear the ABD Bit After Auto-Baud Lock	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
eCAN: When the CAN Option is Invoked in the Boot ROM, the Code may Hang Occasionally	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
eCAN: eCAN-A Boot Mode in Boot ROM	Y	Y	N/A	N/A	N/A	N/A	N/A	N/A	Y	Y	N/A	N/A	Y	Y	N/A	N/A
eCAN: Unexpected Cessation of Transmit Operation	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
eCAN: Abort Acknowledge Bit Not Set	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
WD: Change to Watchdog Module: Bad Key Writes to WDKEY No Longer Cause RESET/Interrupt to be Generated	N/A	N/A	N/A	Y	N/A	N/A	N/A	Y	N/A	N/A	N/A	Y	N/A	N/A	N/A	Y
WD: Limitation on Watchdog Module: Corrupted Watchdog Key Writes	Y	Y	Y	N/A	Y	Y	Y	N/A	Y	Y	Y	N/A	Y	Y	Y	N/A
GPIO: Pin Behavior at Power-up	Y	Y	N/A	N/A	N/A	N/A	N/A	N/A	Y	Y	N/A	N/A	Y	Y	N/A	N/A
GPIO: GPIO Qualification	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Boot ROM: Configuration Change in Boot ROM	Y	Y	N/A	N/A	N/A	N/A	N/A	N/A	Y	Y	N/A	N/A	Y	Y	N/A	N/A
Pulldown Resistor for $\overline{\text{TRST}}$ Pin	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Input Clock: Device Startup Using XCLKIN Input	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
eQEP: Missed First Index Event	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
eQEP: eQEP Inputs in GPIO Asynchronous Mode	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
eQEP: Position Counter Incorrectly Reset on Direction Change During Index	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

(1) Y = Yes; N/A = Not Applicable

Table 5-6. List of Advisories for F28015 and F28016

TITLE	SILICON REVISION(S) AFFECTED(1)							
	F28015				F28016			
	0	A	B	C	0	A	B	C
Memory: Flash and OTP Prefetch Buffer Overflow	Y	Y	Y	Y	Y	Y	Y	Y
Memory: Prefetching Beyond Valid Memory	Y	Y	Y	Y	Y	Y	Y	Y
ADC: Simultaneous Sampling Latency	Y	Y	Y	Y	Y	Y	Y	Y
ADC: ADC Inaccuracy at Low Frequencies	Y	Y	Y	Y	Y	Y	Y	Y
ADC: Initial Conversion Latency	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
ADC: ADC A Channel to B Channel Crosstalk in Simultaneous Mode	Y	Y	Y	N/A	Y	Y	Y	N/A
SCI: Incorrect Operation of SCI in Address Bit Mode	Y	Y	Y	Y	Y	Y	Y	Y
SCI: Bootloader Does Not Clear the ABD Bit After Auto-Baud Lock	Y	Y	Y	Y	Y	Y	Y	Y
eCAN: When the CAN Option is Invoked in the Boot ROM, the Code may Hang Occasionally	Y	Y	Y	Y	Y	Y	Y	Y
eCAN: eCAN-A Boot Mode in Boot ROM	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
eCAN: Unexpected Cessation of Transmit Operation	N/A	N/A	N/A	N/A	Y	Y	Y	Y
eCAN: Abort Acknowledge Bit Not Set	Y	Y	Y	Y	Y	Y	Y	Y
WD: Change to Watchdog Module: Bad Key Writes to WDKEY No Longer Cause RESET/Interrupt to be Generated	N/A	N/A	N/A	Y	N/A	N/A	N/A	Y
WD: Limitation on Watchdog Module: Corrupted Watchdog Key Writes	Y	Y	Y	N/A	Y	Y	Y	N/A
GPIO: Pin Behavior at Power-up	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
GPIO: GPIO Qualification	Y	Y	Y	Y	Y	Y	Y	Y

Table 5-6. List of Advisories for F28015 and F28016 (continued)

TITLE	SILICON REVISION(S) AFFECTED(1)							
	F28015				F28016			
	0	A	B	C	0	A	B	C
Boot ROM: Configuration Change in Boot ROM	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Pulldown Resistor for $\overline{\text{TRST}}$ Pin	Y	Y	Y	Y	Y	Y	Y	Y
Input Clock: Device Startup Using XCLKIN Input	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
eQEP: Missed First Index Event	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
eQEP: eQEP Inputs in GPIO Asynchronous Mode	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
eQEP: Position Counter Incorrectly Reset on Direction Change During Index	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

(1) Y = Yes; N/A = Not Applicable

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

Revision(s) Affected 0, A 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 are 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. In addition, the –flash_prefetch_warn compiler option can be used to issue a warning if the assembly code violates this erratum.

2. Compiler-generated assembly:

Compiler versions prior to V15.12.0.LTS:
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 v4.1.4 and later.

Compiler V15.12.0.LTS and later:
Indicate which functions will run from SARAM using the --ramfunc=on option or the _*attribute*_((ramfunc)). The compiler will only generate SBF/BF instructions within these functions.

The -me switch is deprecated and no longer has any effect.

Advisory	***Memory: Prefetching Beyond Valid Memory***
Revision(s) Affected	0, A 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, A 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: ADC Inaccuracy at Low Frequencies***
Revision(s) Affected	0, A on F2809 silicon 0, A on C280x silicon 0, A, B, C on F2801, F2802, F2806, F2808, and F2801x silicon
Details	At ADCCLK frequencies of less than 1 MHz, the ADC may give inaccurate results on some devices. The inaccuracy will be worse at cold temperature. Small ACQPS settings (less than 3) are more likely to show the inaccuracy.
Workaround(s)	Operate ADCCLK at 1 MHz or above. There is no performance improvement gained by operating the ADCCLK at low frequencies. It is recommended that ADCCLK be set at the maximum value specified in the data sheet or down to one-half the maximum value specified in the data sheet.

Advisory	***ADC: Initial Conversion Latency***
Revision(s) Affected	0 on F2808, F2806, and F2801 silicon
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, and so forth. 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 *TMS320F280x, TMS320C280x, TMS320F2801x DIgital Signal Processors Data Manual* for more information.

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

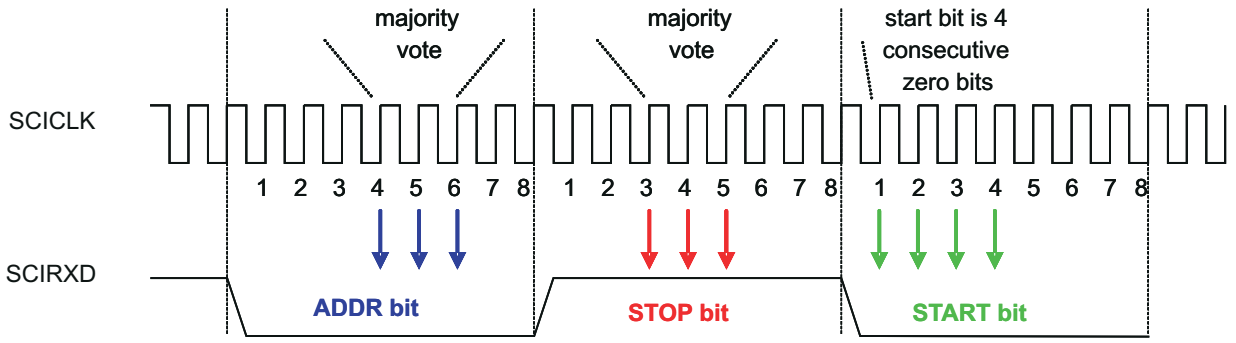
Revision(s) Affected 0, A on F2809 silicon

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:

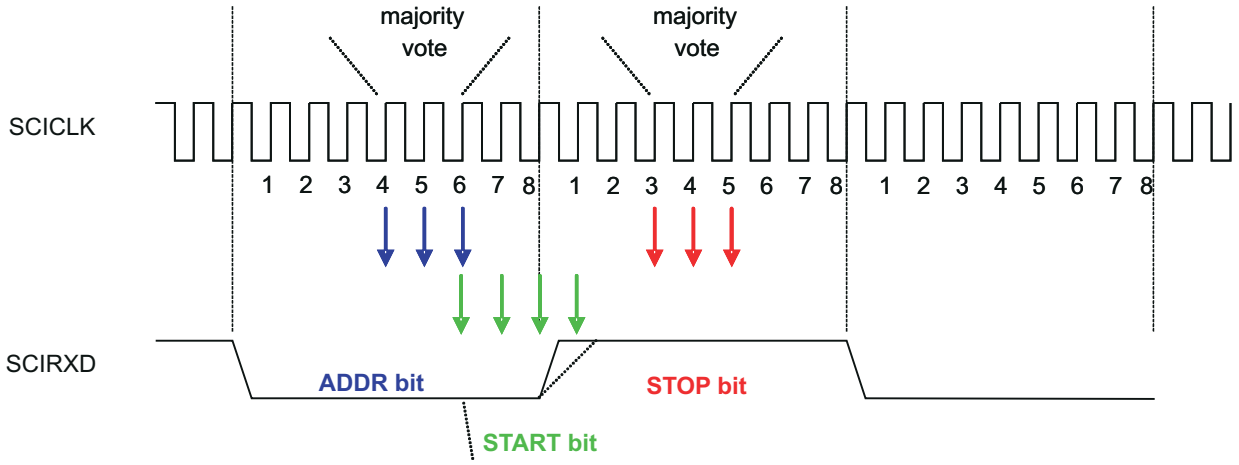


Figure 5-1. Difference Between Expected and Erroneous Operation of START Bit

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, A 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, A on F2809 silicon 0, A on C280x silicon 0, A, B, C on 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, and F2801 silicon
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: Unexpected Cessation of Transmit Operation***
Revision(s) Affected	0, A on F2809 silicon 0, A on C280x silicon 0, A, B, C on F2801, F2802, F2806, F2808, and F28016 silicon
Details	In rare instances, the cessation of message transmission from the eCAN module has been observed (while the receive operation continues normally). This anomalous state may occur without any error frames on the bus.
Workaround(s)	The Time-out feature (MOTO) of the eCAN module may be employed to detect this condition. When this occurs, set and clear the CCR bit (using the CCE bit for verification) to remove the anomalous condition.

Advisory ***eCAN: Abort Acknowledge Bit Not Set***

Revision(s) Affected 0, A 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; that is, 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, A 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* 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* 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, and F2801 silicon
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	***GPIO: GPIO Qualification***
Revision(s) Affected	0, A on F2809 silicon 0, A on C2801 and C2802 silicon 0, A, B, C on F2801, F2802, F2806, F2808, F28015, and F28016 silicon
Details	If a GPIO pin is configured for "n" SYSCLKOUT cycle qualification period (where $1 \le n \le 510$) with "m" qualification samples (m = 3 or 6), it is possible that an input pulse of [n * m – (n – 1)] width may get qualified (instead of n * m). This depends upon the alignment of the asynchronous GPIO input signal with respect to the phase of the internal prescaled clock, and hence, is not deterministic. The probability of this kind of wrong qualification occurring is "1/n". **Worst-case example:** If n = 510, m = 6, a GPIO input width of (n * m) = 3060 SYSCLKOUT cycles is required to pass qualification. However, because of the issue described in this advisory, the minimum GPIO input width which may get qualified is [n * m – (n – 1)] = 3060 – 509 = 2551 SYSCLKOUT cycles.
Workaround(s)	None. Ensure a sufficient margin is in the design for input qualification.

Advisory	***Boot ROM: Configuration Change in Boot ROM***
Revision(s) Affected	0, A on F2808, F2806, and F2801 silicon
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.

Advisory	***Pulldown Resistor for $\overline{\text{TRST}}$ Pin***
Revision(s) Affected	0, A on F2809 silicon 0, A on C280x silicon 0, A, B, C on F2801, F2802, F2806, F2808, and F2801x silicon
Details	In the device data sheet, the recommendation of an external pulldown resistor has now been made a requirement. Earlier, the data sheet suggested leaving this pin unconnected in low-noise environments. Since the term "low-noise" is not easily quantified, an external pulldown resistor has been made a requirement for more robust operation.
Workaround(s)	An external pulldown resistor is required on the $\overline{\text{TRST}}$ pin.

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 has been fixed in revision A of the silicon.

Advisory	***eQEP: Missed First Index Event***
Revision(s) Affected	0, A on F2809 silicon 0, A on C280x silicon 0, A, B, C on F2801, F2802, F2806, and F2808 silicon
Details	If the first index event edge at the QEPI input occurs at any time from one system clock cycle before the corresponding QEPA/QEPB edge to two system clock cycles after the corresponding QEPA/QEP edge, then the eQEP module may miss this index event. This can result in the following behavior: • QPOSCNT will not be reset on the first index event if QEPCTL[PCRM] = 00b or 10b (position counter reset on an index event or position counter reset on the first index event). • The first index event marker flag (QEPSTS[FIMF]) will not be set.
Workaround(s)	Reliable operation is achieved by delaying the index signal such that the QEPI event edge occurs at least two system clock cycles after the corresponding QEPA/QEPB signal edge. For cases where the encoder may impart a negative delay (t_d) to the QEPI signal with respect to the corresponding QEPA/QEPB signal (that is, QEPI edge occurs before the corresponding QEPA/QEPB edge), the QEPI signal should be delayed by an amount greater than "t_d + 2*SYSCLKOUT".

Advisory	***eQEP: eQEP Inputs in GPIO Asynchronous Mode***
Revision(s) Affected	0, A on F2809 silicon 0, A on C280x silicon 0, A, B, C on F2801, F2802, F2806, and F2808 silicon
Details	If any of the eQEP input pins are configured for GPIO asynchronous input mode via the GPxQSELn registers, the eQEP module may not operate properly. For example, QPOSCNT may not reset or latch properly, and pulses on the input pins may be missed. This is because the eQEP peripheral assumes the presence of external synchronization to SYSCLKOUT on inputs to the module. For proper operation of the eQEP module, input GPIO pins should be configured via the GPxQSELn registers for synchronous input mode (with or without qualification). This is the default state of the GPxQSEL registers at reset. All existing eQEP peripheral examples supplied by TI also configure the GPIO inputs for synchronous input mode. The asynchronous mode should not be used for eQEP module input pins.
Workaround(s)	Configure GPIO inputs configured as eQEP pins for non-asynchronous mode (any GPxQSELn register option except “11b = Asynchronous”).

Advisory	***eQEP: Position Counter Incorrectly Reset on Direction Change During Index***
Revision(s) Affected	0, A on F2809 silicon 0, A on C280x silicon 0, A, B, C on F2801, F2802, F2806, and F2808 silicon
Details	While using the PCRM = 0 configuration, if the direction change occurs when the index input is active, the position counter (QPOSCNT) could be reset erroneously, resulting in an unexpected change in the counter value. This could result in a change of up to

±4 counts from the expected value of the position counter and lead to unexpected subsequent setting of the error flags.

While using the PCRM = 0 configuration [that is, Position Counter Reset on Index Event (QEPCTL[PCRM] = 00)], if the index event occurs during the forward movement, then the position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in QEPSTS registers. It also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for index event reset operation.

If the direction change occurs while the index pulse is active, the module would still continue to look for the relative quadrature transition for performing the position counter reset. This results in an unexpected change in the position counter value.

Workaround(s)

Do not use the PCRM = 0 configuration if the direction change could occur while the index is active and the resultant change of the position counter value could affect the application.

Other options for performing position counter reset, if appropriate for the application [such as Index Event Initialization (IEI)], do not have this issue.

6 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 280x DSPs, please see the *TMS320F280x, TMS320C280x, TMS320F2801x Digital Signal Processors Data Manual*.

7 Trademarks

MicroStar BGA™ and TMS320™ are trademarks of Texas Instruments.
All other trademarks are the property of their respective owners.

8 Revision History

Changes from March 18, 2020 to September 30, 2020 (from Revision S (February 2020) to Revision T (September 2020)) **Page**

- Figure 3-2 (Examples of Device Nomenclature): Updated Figure..... 3

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

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