1 Introduction

This document describes the silicon updates to the functional specifications for the TMS320F28004x (F28004x) microcontrollers (MCUs).

The updates are applicable to the following:
- 100-pin Low-Profile Quad Flatpack, PZ Suffix
- 64-pin Low-Profile Quad Flatpack, PM Suffix
- 56-pin Very Thin Quad Flatpack No-Lead, RSH Suffix

2 Device and Development Support Tool Nomenclature

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all TMS320 MCU devices and support tools. Each TMS320™ MCU commercial family member has one of three prefixes: TMX, TMP, or TMS (for example, TMS320F280049). 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 (with TMX for devices and TMDX for tools) through fully qualified production devices and tools (with TMS for devices and TMDS for tools).

- **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, PZ) and temperature range (for example, S).
3 Device Markings

Figure 1 and Figure 2 provide examples of the F28004x device markings and define 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 3 shows an example of the device nomenclature.

![Figure 1. Examples of Device Markings for PM and PZ Packages](image)

YMLLLL = Lot Trace Code

YM = 2-Digit Year/Month Code

LLLL = Assembly Lot

S = Assembly Site Code

980 = TI E.I.A. Code

$$ = Wafer Fab Code (one or two characters) as applicable

# = Silicon Revision Code

G4 = Green (Low Halogen and RoHS-compliant)

Figure 1. Examples of Device Markings for PM and PZ Packages

![Figure 2. Example of Device Markings for RSH Package](image)

YMLLLL = Lot Trace Code

YM = 2-Digit Year/Month Code

LLLL = Assembly Lot

S = Assembly Site Code

$$ = Wafer Fab Code (one or two characters) as applicable

# = Silicon Revision Code

G4 = Green (Low Halogen and RoHS-compliant)

Figure 2. Example of Device Markings for RSH Package

<table>
<thead>
<tr>
<th>SILICON REVISION CODE</th>
<th>SILICON REVISION</th>
<th>REVID&lt;sup&gt;(1)&lt;/sup&gt;</th>
<th>COMMENTS&lt;sup&gt;(2)&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blank</td>
<td>0</td>
<td>0x0000 0000</td>
<td>This silicon revision is available as TMX.</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>0x0000 0001</td>
<td>This silicon revision is available as TMX.</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>0x0000 0002</td>
<td>This silicon revision is available as TMX and TMS.</td>
</tr>
</tbody>
</table>

<sup>(1)</sup> Silicon Revision ID

<sup>(2)</sup> For orderable device numbers, see the PACKAGING INFORMATION table in the TMS320F28004x Microcontrollers Data Manual.
### Device Markings

<table>
<thead>
<tr>
<th>Generic Part Number:</th>
<th>TMS 320 F 280049</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orderable Part Number:</td>
<td>X (blank) F 280049 PZ S</td>
</tr>
<tr>
<td><strong>PREFIX</strong>&lt;sup&gt;(A)&lt;/sup&gt;</td>
<td>TMX (X) = experimental device, TMP (P) = prototype device, TMS (blank) = qualified device</td>
</tr>
<tr>
<td><strong>DEVICE FAMILY</strong></td>
<td>320 = TMS320 MCU Family</td>
</tr>
<tr>
<td><strong>TECHNOLOGY</strong></td>
<td>F = Flash</td>
</tr>
<tr>
<td><strong>TEMPERATURE RANGE</strong></td>
<td>S = −40°C to 125°C (T&lt;sub&gt;j&lt;/sub&gt;)</td>
</tr>
<tr>
<td></td>
<td>Q = −40°C to 125°C (T&lt;sub&gt;a&lt;/sub&gt;)</td>
</tr>
<tr>
<td></td>
<td>(Q refers to AEC Q100 qualification for automotive applications.)</td>
</tr>
</tbody>
</table>

**PACKAGE TYPE**
- 100-Pin PZ Low-Profile Quad Flatpack (LQFP)
- 64-Pin PM LQFP
- 56-Pin RSH Very Thin Quad Flatpack No-Lead (VQFN)

**DEVICE**
- 280049
- 280048
- 280045
- 280041
- 280040
- 280049C
- 280048C
- 280041C
- 280040C

A Prefixes X and P are used in orderable part numbers.

**Figure 3. Example of Device Nomenclature**
4 Usage Notes and Known Design Exceptions to Functional Specifications

4.1 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 2 shows which silicon revision(s) are affected by each usage note.

Table 2. List of Usage Notes

<table>
<thead>
<tr>
<th>TITLE</th>
<th>SILICON REVISION(S) AFFECTED</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIE: Spurious Nested Interrupt After Back-to-Back PIEACK Write and Manual CPU Interrupt Mask Clear</td>
<td>0</td>
</tr>
<tr>
<td>FPU32 and VCU Back-to-Back Memory Accesses</td>
<td>Yes</td>
</tr>
<tr>
<td>Caution While Using Nested Interrupts</td>
<td>Yes</td>
</tr>
</tbody>
</table>

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

Revision(s) Affected: 0, A, B

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.

```c
//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
```
4.1.2 FPU32 and VCU Back-to-Back Memory Accesses

Revision(s) Affected: 0, A, B

This usage note applies when a VCU memory access and an FPU memory access occur back-to-back. There are three cases:

Case 1. Back-to-back memory reads: one read performed by a VCU instruction (VMOV32) and one read performed by an FPU32 instruction (MOV32).

If an R1 pipeline phase stall occurs during the first read, then the second read will latch the wrong data. If the first instruction is not stalled during the R1 pipeline phase, then the second read will occur properly.

The order of the instructions—FPU followed by VCU or VCU followed by FPU—does not matter. The address of the memory location accessed by either read does not matter.

Case 1 Workaround: Insert one instruction between the two back-to-back read instructions. Any instruction, except a VCU or FPU memory read, can be used.

Case 1, Example 1:

```
VMOV32 VR1,mem32 ; VCU memory read
NOP ; Not a FPU/VCU memory read
MOV32 R0H,mem32 ; FPU memory read
```

Case 1, Example 2:

```
VMOV32 VR1,mem32 ; VCU memory read
VMOV32 mem32, VR2 ; VCU memory write
MOV32 R0H,mem32 ; FPU memory read
```

Case 2. Back-to-back memory writes: one write performed by a VCU instruction (VMOV32) and one write performed by an FPU instruction (MOV32).

If a pipeline stall occurs during the first write, then the second write can corrupt the data. If the first instruction is not stalled in the write phase, then no corruption will occur.

The order of the instructions—FPU followed by VCU or VCU followed by FPU—does not matter. The address of the memory location accessed by either write does not matter.

Case 2 Workaround: Insert two instructions between the back-to-back VCU and FPU writes. Any instructions, except VCU or FPU memory writes, can be used.

Case 2, Example 1:

```
VMOV32 mem32,VR0 ; VCU memory write
NOP ; Not a FPU/VCU memory write
NOP ; Not a FPU/VCU memory write
MOV32 mem32,R3H ; FPU memory write
```

Case 2, Example 2:

```
VMOV32 mem32,VR0 ; VCU memory write
VMOV32 VR1, mem32 ; VCU memory read
NOP
MOV32 mem32,R3H ; FPU memory write
```

Case 3. Back-to-back memory writes followed by a read or a memory read followed by a write. In this case, there is no interaction between the two instructions. No action is required.

Workaround: See Case 1 Workaround and Case 2 Workaround.
4.1.3 Caution While Using Nested Interrupts

**Revision(s) Affected:** 0, A, B

If the user is enabling interrupts using the EINT instruction inside an interrupt service routine (ISR) in order to use the nesting feature, then the user must disable the interrupts before exiting the ISR. Failing to do so may cause undefined behavior of CPU execution.
## 4.2 Known Design Exceptions to Functional Specifications

### Table 3. Table of Contents for Advisories

<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advisory — FPU: FPU-to-CPU Register Move Operation Preceded by Any FPU 2p Operation</td>
<td>9</td>
</tr>
<tr>
<td>Advisory — SDFM: Dynamically Changing Threshold Settings (LLT, HLT), Filter Type, or COSR Settings Will Trigger Spurious Comparator Events</td>
<td>12</td>
</tr>
<tr>
<td>Advisory — SDFM: Dynamically Changing Data Filter Settings (Such as Filter Type or DOSR) Will Trigger Spurious Data Acknowledge Events</td>
<td>12</td>
</tr>
<tr>
<td>Advisory — eQEP: Position Counter Incorrectly Reset on Direction Change During Index</td>
<td>13</td>
</tr>
<tr>
<td>Advisory — eQEP: eQEP Inputs in GPIO Asynchronous Mode</td>
<td>13</td>
</tr>
<tr>
<td>Advisory — V_{DD} Supply: During V_{DDIO} Power Up, V_{DD} May Also Rise</td>
<td>14</td>
</tr>
<tr>
<td>Advisory — eCAP: HRFRC is Not EALLOW-Protected</td>
<td>15</td>
</tr>
<tr>
<td>Advisory — PLL: PLL May Not Lock on the First Lock Attempt</td>
<td>16</td>
</tr>
<tr>
<td>Advisory — LPM: STANDBY Low-Power Mode is Not Supported</td>
<td>17</td>
</tr>
<tr>
<td>Advisory — DCC: Single-Shot-Mode Operation May End Prematurely</td>
<td>18</td>
</tr>
<tr>
<td>Advisory — BOR: VDDIO Between 2.45 V and 3.0 V can Result in Multiple XRSn Pulses</td>
<td>19</td>
</tr>
<tr>
<td>Advisory — I2C: SDA and SCL Open-Drain Output Buffer Issue</td>
<td>20</td>
</tr>
<tr>
<td>Advisory — ePWM: An ePWM Glitch can Occur if a Trip Remains Active at the End of the Blanking Window</td>
<td>22</td>
</tr>
<tr>
<td>Advisory — INTOSC: VDDIO Powered Without VDD Can Cause INTOSC Frequency Drift</td>
<td>23</td>
</tr>
<tr>
<td>Advisory — FSI: RX FIFO Spurious Overrun</td>
<td>24</td>
</tr>
<tr>
<td>Advisory — Boot ROM: Calling SCI Bootloader from Application</td>
<td>25</td>
</tr>
<tr>
<td>Advisory — Memory: Prefetching Beyond Valid Memory</td>
<td>26</td>
</tr>
<tr>
<td>Advisory — ADC: Interrupts may Stop if INTxCONT (Continue-to-Interrupt Mode) is not Set</td>
<td>27</td>
</tr>
<tr>
<td>Advisory — ADC: Degraded ADC Performance With ADCCLK Fractional Divider</td>
<td>28</td>
</tr>
<tr>
<td>Advisory — Analog Subsystem: Software Configuration for Shared Reference Pins</td>
<td>28</td>
</tr>
<tr>
<td>Advisory — Analog Trim of Some TMX Devices</td>
<td>29</td>
</tr>
<tr>
<td>Advisory — PGA: Output Filter Path is Not Supported</td>
<td>30</td>
</tr>
<tr>
<td>Advisory — ROM: Flash API Library and FPU32 Twiddle Factor RFFT Table Not Present</td>
<td>31</td>
</tr>
<tr>
<td>Advisory — REVID: Some TMX Revision A Devices Have an Incorrect REVID Value</td>
<td>32</td>
</tr>
<tr>
<td>Advisory — GPIO: X2/GPIO18 Pin Pullup Current During Power Up</td>
<td>33</td>
</tr>
<tr>
<td>Advisory — GPIO: Open-Drain Configuration May Drive a Short High Pulse</td>
<td>34</td>
</tr>
<tr>
<td>Advisory — GPIO: Parasitic Path to V_{SS} When Maximum V_{IH} is Exceeded in Input Mode</td>
<td>35</td>
</tr>
<tr>
<td>Advisory — GPIO: Pins may Drive High During Power Up</td>
<td>36</td>
</tr>
<tr>
<td>Advisory — GPIO: Signal Latch-up to V_{SS}</td>
<td>37</td>
</tr>
</tbody>
</table>
Table 4 shows which silicon revision(s) are affected by each advisory.

### Table 4. List of Advisories

<table>
<thead>
<tr>
<th>TITLE</th>
<th>SILICON REVISION(S) AFFECTED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>FPU: FPU-to-CPU Register Move Operation Preceded by Any FPU 2p Operation</td>
<td>Yes</td>
</tr>
<tr>
<td>SDFM: Dynamically Changing Threshold Settings (LLT, HLT), Filter Type, or COSR Settings Will Trigger Spurious Comparator Events</td>
<td>Yes</td>
</tr>
<tr>
<td>SDFM: Dynamically Changing Data Filter Settings (Such as Filter Type or DOSR) Will Trigger Spurious Data Acknowledge Events</td>
<td>Yes</td>
</tr>
<tr>
<td>eQEP: Position Counter Incorrectly Reset on Direction Change During Index</td>
<td>Yes</td>
</tr>
<tr>
<td>eQEP: eQEP Inputs in GPIO Asynchronous Mode</td>
<td>Yes</td>
</tr>
<tr>
<td>VDD Supply: During VDDIO Power Up, VDD May Also Rise</td>
<td>Yes</td>
</tr>
<tr>
<td>eCAP: HRFRC is Not EALLOW-Protected</td>
<td>Yes</td>
</tr>
<tr>
<td>PLL: PLL May Not Lock on the First Lock Attempt</td>
<td>Yes</td>
</tr>
<tr>
<td>LPDM: STANDBY Low-Power Mode is Not Supported</td>
<td>Yes</td>
</tr>
<tr>
<td>DCC: Single-Shot-Mode Operation May End Prematurely</td>
<td>Yes</td>
</tr>
<tr>
<td>BOR: VDDIO Between 2.45 V and 3.0 V can Result in Multiple XRSn Pulses</td>
<td>Yes</td>
</tr>
<tr>
<td>I2C: SDA and SCL Open-Drain Output Buffer Issue</td>
<td>Yes</td>
</tr>
<tr>
<td>ePWM: An ePWM Glitch can Occur if a Trip Remains Active at the End of the Blanking Window</td>
<td>Yes</td>
</tr>
<tr>
<td>INTOSC: VDDIO Powered Without VDD Can Cause INTOSC Frequency Drift</td>
<td>Yes</td>
</tr>
<tr>
<td>FSI: RX FIFO Spurious Overrun</td>
<td>Yes</td>
</tr>
<tr>
<td>Boot ROM: Calling SCI Bootloader from Application</td>
<td>Yes</td>
</tr>
<tr>
<td>Memory: Prefetching Beyond Valid Memory</td>
<td>Yes</td>
</tr>
<tr>
<td>ADC: Interrupts may Stop if INTxCONT (Continue-to-Interrupt Mode) is not Set</td>
<td>Yes</td>
</tr>
<tr>
<td>ADC: Degraded ADC Performance With ADCCCLK Fractional Divider</td>
<td>Yes</td>
</tr>
<tr>
<td>Analog Subsystem: Software Configuration for Shared Reference Pins</td>
<td>Yes</td>
</tr>
<tr>
<td>Analog Trim of Some TMX Devices</td>
<td>Yes</td>
</tr>
<tr>
<td>PGA: Output Filter Path is Not Supported</td>
<td>Yes</td>
</tr>
<tr>
<td>ROM: Flash API Library and FPU32 Twiddle Factor RFFT Table Not Present</td>
<td>Yes</td>
</tr>
<tr>
<td>REVID: Some TMX Revision A Devices Have an Incorrect REVID Value</td>
<td>–</td>
</tr>
<tr>
<td>GPIO: X2/GPIO18 Pin Pullup Current During Power Up</td>
<td>Yes</td>
</tr>
<tr>
<td>GPIO: Open-Drain Configuration May Drive a Short High Pulse</td>
<td>Yes</td>
</tr>
<tr>
<td>GPIO: Parasitic Path to VSS When Maximum VIN is Exceeded in Input Mode</td>
<td>–</td>
</tr>
<tr>
<td>GPIO: Pins may Drive High During Power Up</td>
<td>Yes</td>
</tr>
<tr>
<td>GPIO: Signal Latch-up to VSS</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Advisory

FPU: FPU-to-CPU Register Move Operation Preceded by Any FPU 2p Operation

Revision(s) Affected

0, A, B

Details

This advisory applies when a multi-cycle (2p) FPU instruction is followed by a FPU-to-CPU register transfer. If the FPU-to-CPU read instruction source register is the same as the 2p instruction destination, then the read may be of the value of the FPU register before the 2p instruction completes. This occurs because the 2p instructions rely on data-forwarding of the result during the E3 phase of the pipeline. If a pipeline stall happens to occur in the E3 phase, the result does not get forwarded in time for the read instruction.

The 2p instructions impacted by this advisory are MPYF32, ADDF32, SUBF32, and MACF32. The destination of the FPU register read must be a CPU register (ACC, P, T, XAR0...XAR7). This advisory does not apply if the register read is a FPU-to-FPU register transfer.

In the example below, the 2p instruction, MPYF32, uses R6H as its destination. The FPU register read, MOV32, uses the same register, R6H, as its source, and a CPU register as the destination. If a stall occurs in the E3 pipeline phase, then MOV32 will read the value of R6H before the MPYF32 instruction completes.

Example of Problem:

```
MPYF32 R6H, R5H, R0H ; 2p FPU instruction that writes to R6H
|| MOV32 *XAR7++, R4H
F32TOUI16R R3H, R4H ; delay slot
ADDF32 R2H, R2H, R0H
|| MOV32 *--SP, R2H ; alignment cycle
MOV32 @XAR3, R6H ; FPU register read of R6H
```

Figure 4 shows the pipeline diagram of the issue when there are no stalls in the pipeline.

---

**Figure 4. Pipeline Diagram of the Issue When There are no Stalls in the Pipeline**
Figure 5 shows the pipeline diagram of the issue if there is a stall in the E3 slot of the instruction I1.

### Workaround(s)

Treat MPYF32, ADDF32, SUBF32, and MACF32 in this scenario as 3p-cycle instructions. Three NOPs or non-conflicting instructions must be placed in the delay slot of the instruction.

The C28x Code Generation Tools v.6.2.0 and later will both generate the correct instruction sequence and detect the error in assembly code. In previous versions, v6.0.5 (for the 6.0.x branch) and v6.1.2 (for the 6.1.x branch), the compiler will generate the correct instruction sequence but the assembler will not detect the error in assembly code.

### Example of Workaround:

```
MPYF32 R6H, R5H, R0H
|| MOV32 *XAR7++, R4H ; 3p FPU instruction that writes to R6H
F32TOU16R R3H, R4H ; delay slot
ADDF32 R2H, R2H, R0H
|| MOV32 *--SP, R2H ; delay slot
NOP ; alignment cycle
MOV32 @XAR3, R6H ; FPU register read of R6H
```

Figure 6 shows the pipeline diagram with the workaround in place.
<table>
<thead>
<tr>
<th>Instruction</th>
<th>F1</th>
<th>F2</th>
<th>D1</th>
<th>D2</th>
<th>R1</th>
<th>R2</th>
<th>E</th>
<th>W</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPYF32 R6H, R5H, R0H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV32 *XAR7++, R4H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV32 *XAR7++, R4H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDF32 R3H, R2H, R0H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDF32 R3H, R2H, R0H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV32 @XAR3, R6H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 6. Pipeline Diagram With Workaround in Place**
### Advisory

**SDFM: Dynamically Changing Threshold Settings (LLT, HLT), Filter Type, or COSR Settings Will Trigger Spurious Comparator Events**

**Revision(s) Affected**

| 0, A, B |

**Details**

When SDFM comparator settings—such as filter type, lower/upper threshold, or comparator OSR (COSR) settings—are dynamically changed during run time, spurious comparator events will be triggered. The spurious comparator event will trigger a corresponding CPU interrupt, CLA task, ePWM X-BAR events, and GPIO output X-BAR events if configured appropriately.

**Workaround(s)**

When comparator settings need to be changed dynamically, follow the procedure below to ensure spurious comparator events do not generate a CPU interrupt, CLA task, or X-BAR events (ePWM X-BAR/GPIO output X-BAR events):

1. Disable the SDFM comparator interrupt.
2. Change comparator settings such as lower/upper threshold, filter type, or COSR.
3. Delay for at least a latency of comparator filter + 5 SD-Cx clock cycles.
4. Enable the SDFM comparator interrupt.

### Advisory

**SDFM: Dynamically Changing Data Filter Settings (Such as Filter Type or DOSR) Will Trigger Spurious Data Acknowledge Events**

**Revision(s) Affected**

| 0, A, B |

**Details**

When SDFM data settings—such as filter type or DOSR settings—are dynamically changed during run time, spurious data-filter-ready events will be triggered. The spurious data-ready event will trigger a corresponding CPU interrupt, CLA task, and DMA trigger if configured appropriately.

**Workaround(s)**

When SDFM data filter settings need to be changed dynamically, follow the procedure below to ensure spurious data-filter-ready events are not generated:

1. Disable the SDFM data filter.
2. Change SDFM data filter settings such as filter type or DOSR.
3. Delay for at least a latency of data filter + 5 SD-Cx clock cycles.
4. Enable the SDFM data filter.
Advisory  

**eQEP: Position Counter Incorrectly Reset on Direction Change During Index**

**Revision(s) Affected**  
0, A, B

**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.

The next index event without a simultaneous direction change will reset the counter properly and work as expected.

**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.

---

Advisory  

**eQEP: eQEP Inputs in GPIO Asynchronous Mode**

**Revision(s) Affected**  
0, A, B

**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 because the eQEP peripheral assumes the presence of external synchronization to SYSCLKOUT on inputs to the module. For example, QPOS_CNT may not reset or latch properly, and pulses on the input pins may be missed.

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), which 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**

| V<sub>DD</sub> Supply: During V<sub>DDIO</sub> Power Up, V<sub>DD</sub> May Also Rise |

**Revision(s) Affected**

0, A, B

**Details**

A leakage current from V<sub>DDIO</sub> to V<sub>DD</sub> is present when the V<sub>DD</sub> supply is below approximately 0.5 V. This causes the V<sub>DD</sub> voltage to rise to approximately 0.5 V when V<sub>DDIO</sub> is powered. This is observed when the device is configured to use either the internal VREG (VREGENZ tied to V<sub>SS</sub>) or an external 1.2-V regulator (VREGENZ tied to V<sub>DDIO</sub>), and there is a significant delay (about 1 ms) between the power up of V<sub>DDIO</sub> and V<sub>DD</sub> from external regulators or the ramp time of V<sub>DDIO</sub> is greater than 1 ms when in internal VREG mode.

This does not impact device functionality once the external 1.2-V or internal 1.2-V supply begins to ramp. See the TMS320F28004x Microcontrollers Data Manual for power sequencing requirements.

**Workaround(s)**

If this early voltage on V<sub>DD</sub> is a problem for system-level supervisor circuits, then minimize the delay between ramping the 3.3-V V<sub>DDIO</sub> and 1.2-V V<sub>DD</sub> rails. If the internal VREG is used, decrease the ramp time of the 3.3-V V<sub>DDIO</sub> supply to 1 ms or less.
### Advisory: eCAP: HRFRC is Not EALLOW-Protected

<table>
<thead>
<tr>
<th>Revision(s) Affected</th>
<th>Details</th>
<th>Workaround(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, A, B</td>
<td>The HRFRC register is not EALLOW-protected. Issuing the EALLOW and EDIS instructions to write to this register is not required. To enable software reuse on other devices where HRFRC is EALLOW-protected, using EALLOW and EDIS is recommended.</td>
<td>None</td>
</tr>
</tbody>
</table>
## Advisory

### PLL: PLL May Not Lock on the First Lock Attempt

#### Revision(s) Affected

0, A, B

#### Details

The PLL may not start properly at device power up. The PLLSTS[LOCKS] bit is set, but the PLL does not produce a clock.

Once the PLL has started properly, the PLL can be disabled and reenabled with no issues and will stay locked. However, the PLL lock problem could reoccur on a subsequent power-up cycle.

If the SYSPLL has not started properly and is selected as the CPU clock source, the CPU will stop executing instructions.

The occurrence rate of this transient issue is low. After an initial occurrence, this issue may not be subsequently observed in the system again. Implementation of the workaround reduces the rate of occurrence.

#### Workaround(s)

TI recommends doing lock sequences in succession until the PLL is in locked state when the PLL is configured for the first time after power up. The lock sequence is: disable the PLL, start the PLL, wait for the LOCKS bit to set, and validate the PLL frequency using the Dual Clock Comparator (DCC). After the PLL is observed to be running, it can be selected as the CPU clock source.

TI recommends using the C2000Ware SysCtl_setClock() function, which also includes implementation of this workaround, to set the PLL clock.

Details on DCC usage are in the C2000Ware SysCtl_IsPLLValid() function.

The workaround can also be applied at the system level by a supervisor resetting the device if it is not responding.
### Advisory

**LPM: STANDBY Low-Power Mode is Not Supported**

<table>
<thead>
<tr>
<th>Revision(s) Affected</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, A, B</td>
<td>The STANDBY low-power mode is not supported.</td>
</tr>
</tbody>
</table>

**Workaround(s)**

The IDLE or HALT low-power modes can be used for power reduction. See the *TMS320F28004x Microcontrollers Technical Reference Manual* for information on implementing these modes.

If IDLE is used, additional power reduction can be optionally achieved through software by one or all of these methods:

- Decrease the SYSCLK frequency:
  - Change the SYSCLK source to OSCCLK by configuring SYSPLLCTL1[PLLCLKEN] = 0.
  - Change the SYSCLKDIVSEL register to a higher divider.
- Disable peripheral clocks through the PCLKCRx register.
**Advisory**

<table>
<thead>
<tr>
<th>DCC: Single-Shot-Mode Operation May End Prematurely</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Revision(s) Affected</strong></td>
</tr>
<tr>
<td><strong>Details</strong></td>
</tr>
</tbody>
</table>
| **Workaround(s)** | Any of the following conditions ends DCC operation prematurely. TI recommends rerunning DCC if any of the below conditions are met.  
• DCCSTATUS[DONE] = 1 and (DCCCNT1 > 0 or DCCCNT0 > 0 or DCCVALID0 > 0)  
• DCCSTATUS[ERROR] = 1 and DCCCNT1 > 0 and DCCVALID0 > 0 |
Advisory

BOR: VDDIO Between 2.45 V and 3.0 V can Result in Multiple XRSn Pulses

Revision(s) Affected
0, A, B

Details
The BOR can generate repeating XRSn assertions and deassertions when the VDDIO supply voltage is between 2.45 V and 3.0 V. It is recommended that the XRSn pin not be used directly as a reset to any other devices in the system.

The F28004x BOR is effective for internally holding the device in a known reset state, even when these XRSn pulses are occurring. The device will not branch to application code or bootloaders, and all other pins will be held in their reset state until the VDDIO supply rises above 3.0 V.

Workaround(s)
1. Ignore the extra XRSn transitions during power up, power down, and BOR events. The extra XRSn pulses will have no effect on the F28004x device operation itself.
2. If XRSn pulses would cause undesired system behavior with other system components, then do not use XRSn to drive other devices. An external voltage supervisor can be used for these applications.
3. For applications that need to avoid these pulses during normal power up and power down:
   a. Power up: Follow the t_VDDIO-RAMP requirement in the Recommended Operating Conditions table of the TMS320F28004x Microcontrollers Data Manual; no extra XRSn low pulses will occur.
   b. Power Down: To avoid any deassertion of XRSn during power down, design the power supply so that VDDIO passes through the range from 3.0 V to 2.45 V within 25 µs. If some voltage rise on XRSn is acceptable, then the time constant of the RC circuit implemented on XRSn can be calculated to ensure the voltage does not rise above a system-specified threshold.
Advisory

I2C: SDA and SCL Open-Drain Output Buffer Issue

Revision(s) Affected
0, A, B

Details
The SDA and SCL outputs are implemented with push-pull 3-state output buffers rather than open-drain output buffers as required by I2C. While it is possible for the push-pull 3-state output buffers to behave as open-drain outputs, an internal timing skew issue causes the outputs to drive a logic-high for a duration of 0–5 ns before the outputs are disabled. The unexpected high-level pulse will only occur when the SCL or SDA outputs transition from a driven low state to a high-impedance state and there is sufficient internal timing skew on the respective I2C output.

This short high-level pulse injects energy in the I2C signals traces, which causes the I2C signals to sustain a period of ringing as a result of multiple transmission line reflections. This ringing should not cause an issue on the SDA signal because it only occurs at times when SDA is expected to be changing logic levels and the ringing will have time to damp before data is latched by the receiving device. The ringing may have enough amplitude to cross the SCL input buffer switching threshold several times during the first few nanoseconds of this ringing period, which may cause clock glitches. This ringing should not cause a problem if the amplitude is damped within the first 50 ns because I2C devices are required to filter their SCL inputs to remove clock glitches. Therefore, it is important to design the PCB signal traces to limit the duration of the ringing to less than 50 ns. One possible solution is to insert series termination resistors near the SCL and SDA terminals to attenuate transmission line reflections.

This issue may also cause the SDA output to be in contention with the slave SDA output for the duration of the unexpected high-level pulse when the slave begins its ACK cycle. This occurs because the slave may already be driving SDA low before the unexpected high-level pulse occurs. The glitch that occurs on SDA as a result of this short period of contention does not cause any I2C protocol issue but the peak current applies unwanted stress to both I2C devices and potentially increases power supply noise. Therefore, a series termination resistor located near the respective SDA terminal is required to limit the current during the short period of contention.

A similar contention problem can occur on SCL when connected to I2C slave devices that support clock stretching. This occurs because the slave is driving SCL low before the unexpected high-level pulse occurs. The glitch that occurs on SCL as a result of this short period of contention does not cause any I2C protocol issue because I2C devices are required to apply a glitch filter to their SCL inputs. However, the peak current applies unwanted stress to both I2C devices and potentially increases power supply noise. Therefore, a series termination resistor located near the respective SCL terminal is required to limit the current during the short period of contention.

If another master is connected, the unexpected high-level pulses on the SCL and SDA outputs can cause contention during clock synchronization and arbitration. The series termination resistors described above will also limit the contention current in this use case without creating any I2C protocol issue.

Workaround(s)
Insert series termination resistors on the SCL and SDA signals and locate them near the SCL and SDA terminals. The SCL and SDA pullup resistors should also be located near the SCL and SDA terminals. The placement of the series termination resistor and pullup resistor should be connected as shown in Figure 7.
Figure 7. Placement of Series Termination Resistor and Pullup Resistor

Table 5 provides series termination and pullup resistor value recommendations. The I2C signal level and respective VDDIO power supply voltage is shown in the first column. Two resistor value combination options are provided for each voltage. One option supports a maximum high-level input current of 200 µA to all attached I2C devices, while the other option supports a maximum high-level input current of 100 µA to all attached I2C devices.

Table 5. Recommended Values for Series Termination Resistor and Pullup Resistor

<table>
<thead>
<tr>
<th>I2C SIGNAL LEVEL AND RESPECTIVE VDDIO POWER SUPPLY (V)</th>
<th>SERIES TERMINATION RESISTOR (Ω)</th>
<th>PULLUP RESISTOR (Ω)</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3</td>
<td>60</td>
<td>3300</td>
<td>Maximum high-level input current up to 200 µA</td>
</tr>
<tr>
<td>3.3</td>
<td>75</td>
<td>6600</td>
<td>Maximum high-level input current up to 100 µA</td>
</tr>
</tbody>
</table>
Advisory  

**ePWM: An ePWM Glitch can Occur if a Trip Remains Active at the End of the Blanking Window**

**Revision(s) Affected**  0, A, B

**Details**  The blanking window is typically used to mask any PWM trip events during transitions which would be false trips to the system. If an ePWM trip event remains active for less than three ePWM clocks after the end of the blanking window cycles, there can be an undesired glitch at the ePWM output.

Figure 8 illustrates the time period which could result in an undesired ePWM output.

![Figure 8. Undesired Trip Event and Blanking Window Expiration](image)

**Figure 9. Resulting Undesired ePWM Outputs Possible**

**Workaround(s)**  Extend or reduce the blanking window to avoid any undesired trip action.
<table>
<thead>
<tr>
<th>Advisory</th>
<th>INTOSC: VDDIO Powered Without VDD Can Cause INTOSC Frequency Drift</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revision(s) Affected</td>
<td>0, A, B</td>
</tr>
</tbody>
</table>
| Details | The "D" revision of the *TMS320F28004x Microcontrollers Data Manual* (SPRS945D) has updated power sequencing requirements. Revision "C" and earlier revisions of the data manual did not require VDDIO and VDD to be powered on and powered off at the same time when using an external supply source for VDD.  
If VDDIO is powered on while VDD is not powered, there will be an accumulating and persistent downward frequency drift for INTOSC1 and INTOSC2. The rate of drift accumulated will be greater when VDDIO is powered without VDD at high temperatures.  
As a result of this drift, the INTOSC1 and INTOSC2 internal oscillator frequencies could fall below the minimum values specified in the data manual. This would impact applications using INTOSC2 as the clock source for the PLL, with the system operating at a lower frequency than expected. |
| Workaround(s) | 1. Use the internal VREG or internal DCDC, which will ensure VDD is powered when VDDIO is present.  
2. When using an external VDD source, always keep VDDIO and VDD powered together.  
3. Use the external X1 and X2 crystal oscillators as the PLL clock source. The crystal oscillator does not have any drift related to VDDIO and VDD supply sequencing. |
### Advisory

**FSI: RX FIFO Spurious Overrun**

<table>
<thead>
<tr>
<th>Revision(s) Affected</th>
<th>Details</th>
<th>Workaround(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, A, B</td>
<td>A buffer overrun is asserted when the last location of the FIFO is written.</td>
<td>Two possible workarounds are available.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. Set up the communication between the transmitting and receiving modules in such a way that the maximum number of data words received, before the first data word is read, is 15 (not 16). Under this condition, buffer overrun behavior is reliable.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. If the application must fill all 16 data words in the receive buffer before the first data word is read (NWORD packet with 16 words), then the following sequence can be used:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Ignore RX buffer overrun RX_EVT_STS.BUF_OVERRUN.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• On RX_EVT_STS.FRAME_DONE, read RX_BUF_PTR_STS.CURR_WORD_CNT and check that it is 16.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Use DMA or software to move the data out of the RX buffer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Read RX_BUF_PTR_STS.CURR_WORD_CNT and check that it is 0.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Clear the RX_EVT_STS.FRAME_DONE Flag by writing a 1 to RX_EVT_CLR.FRAME_DONE.</td>
</tr>
<tr>
<td>Advisory</td>
<td><strong>Boot ROM: Calling SCI Bootloader from Application</strong></td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Revision(s) Affected</strong></td>
<td>0, A, B</td>
<td></td>
</tr>
<tr>
<td><strong>Details</strong></td>
<td>The ROM SCI bootloader uses autobaud lock to lock the baud rate. The SCI baud rate is split between two registers, SCILBAUD and SCIHBAUD. The ROM SCI bootloader expects SCIHBAUD to contain its default reset value of zero. If the ROM SCI bootloader is called from an application that modified the contents of SCIHBAUD to be non-zero, then the SCI will not autobaud-lock and the SCI bootloader will not execute.</td>
<td></td>
</tr>
<tr>
<td><strong>Workaround(s)</strong></td>
<td>Clear SCIHBAUD to zero before calling the ROM SCI Bootloader.</td>
<td></td>
</tr>
</tbody>
</table>
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 | M1, GS3 – The prefetch queue is 8 x16 words in depth. Therefore, code should not come within 8 words of the end of valid memory. Prefetching across the boundary between two valid memory blocks is all right.

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.

Flash – The prefetch queue is 16 x16 words in depth. Therefore, code should not come within 16 words of the end of valid memory; otherwise, it generates a Flash ECC uncorrectable error.

<table>
<thead>
<tr>
<th>MEMORY TYPE</th>
<th>ADDRESSES IMPACTED</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>0x0000 07F8–0x0000 07FF</td>
</tr>
<tr>
<td>GS3</td>
<td>0x0001 3FF8–0x0001 3FFF</td>
</tr>
<tr>
<td>Flash</td>
<td>0x0009 FFF0–0x0009 FFFF</td>
</tr>
</tbody>
</table>
**Advisory**

**ADC: Interrupts may Stop if INTxCONT (Continue-to-Interrupt Mode) is not Set**

**Revision(s) Affected**
0, A, B

**Details**

If ADCINTSELxNx[INTxCONT] = 0, then interrupts will stop when the ADCINTFLG is set and no additional ADC interrupts will occur.

When an ADC interrupt occurs simultaneously with a software write of the ADCINTFLGCLR register, the ADCINTFLG will unexpectedly remain set, blocking future ADC interrupts.

**Workaround(s)**

1. Use Continue-to-Interrupt Mode to prevent the ADCINTFLG from blocking additional ADC interrupts:

   ```
   ADCINTSEL1N2[INT1CONT] = 1;
   ADCINTSEL1N2[INT2CONT] = 1;
   ADCINTSEL3N4[INT3CONT] = 1;
   ADCINTSEL3N4[INT4CONT] = 1;
   ```

2. Ensure there is always sufficient time to service the ADC ISR and clear the ADCINTFLG before the next ADC interrupt occurs to avoid this condition.

3. Check for an overflow condition in the ISR when clearing the ADCINTFLG. Check ADCINTOVF immediately after writing to ADCINTFLGCLR; if it is set, then write ADCINTFLGCLR a second time to ensure the ADCINTFLG is cleared. The ADCINTOVF register will be set, indicating an ADC conversion interrupt was lost.

   ```
   AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1; //clear INT1 flag
   if(1 == AdcaRegs.ADCINTOVF.bit.ADCINT1) //ADCINT overflow
   {
     AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1; //clear INT1 again
     // If the ADCINTOVF condition will be ignored by the application
     // then clear the flag here by writing 1 to ADCINTOVFCLR.
     // If there is a ADCINTOVF handling routine, then either insert
     // that code and clear the ADCINTOVF flag here or do not clear
     // the ADCINTOVF here so the external routine will detect the
     // condition.
     // AdcaRegs.ADCINTOVFCLR.bit.ADCINT1 = 1; // clear OVF
   }
   ```
Advisory

**ADC: Degraded ADC Performance With ADCCLK Fractional Divider**

<table>
<thead>
<tr>
<th>Revision(s) Affected</th>
<th>0, A, B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Details</td>
<td>Using fractional SYSCLK-to-ADCCLK dividers (controlled by the ADCCTL2.PRESCALE field) has been shown to cause degradation in ADC performance on this device. See Table 7.</td>
</tr>
</tbody>
</table>

### Table 7. ADCCTL2 Register

<table>
<thead>
<tr>
<th>REDUCED PERFORMANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT</td>
</tr>
<tr>
<td>3–0</td>
</tr>
<tr>
<td>3–0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NORMAL PERFORMANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT</td>
</tr>
<tr>
<td>3–0</td>
</tr>
<tr>
<td>3–0</td>
</tr>
</tbody>
</table>

**Workaround(s)**

Use even PRESCALE clock divider values. Even PRESCALE values result in integer clock dividers which do not impact the ADC performance.

Advisory

**Analog Subsystem: Software Configuration for Shared Reference Pins**

<table>
<thead>
<tr>
<th>Revision(s) Affected</th>
<th>0, A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Details</td>
<td>Smaller pin-count packages of the F28004x device family have combined VREFHI pins. Software configuration bits are provided in the ANAREFPP register to disable all but one of the ganged references. This allows correct operation of internal reference mode in these circumstances. On production (TMS) devices, the Boot ROM will write these bits, and no further action will be required from the user. However, on some TMX devices, this write will not occur.</td>
</tr>
</tbody>
</table>

**Workaround(s)**

For TMX devices, the user should do the following writes one time before trying to configure the references for internal reference mode:

- 100-pin PZ package: The value 0x0002 should be written to ANAREFPP.
- 64-pin PM package: The value 0x0003 should be written to ANAREFPP.
- 56-pin RSH package: The value 0x0003 should be written to ANAREFPP.
Advisory  

**Analog Trim of Some TMX Devices**

**Revision(s) Affected**  
0, A

**Details**  
Some TMX samples may not have analog trims programmed. This could degrade the performance of the ADC, buffered DAC, internal oscillators, PGA, and internal voltage regulator. A value of all zeros in these trim registers will have the following impact.

<table>
<thead>
<tr>
<th>TRIM</th>
<th>REGISTER</th>
<th>IMPACT OF TRIM VALUE EQUAL TO ZERO</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC offset</td>
<td>AdcaRegs.ADCOFFTRIM</td>
<td>Degraded performance of the ADC offset error specification.</td>
</tr>
<tr>
<td></td>
<td>AdcbRegs.ADCOFFTRIM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AdccRegs.ADCOFFTRIM</td>
<td></td>
</tr>
<tr>
<td>ADC reference</td>
<td>AnalogSubsysRegs.ANAREFTRIMA</td>
<td>Degraded performance of the ADC for all specifications. No workaround available.</td>
</tr>
<tr>
<td></td>
<td>AnalogSubsysRegs.ANAREFTRIMB</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AnalogSubsysRegs.ANAREFTRIMC</td>
<td></td>
</tr>
<tr>
<td>ADC linearity</td>
<td>AdcaRegs.ADCINLTRIM2-3</td>
<td>Degraded INL and DNL specifications of the ADC. No workaround available.</td>
</tr>
<tr>
<td></td>
<td>AdcbRegs.ADCINLTRIM2-3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AdccRegs.ADCINLTRIM2-3</td>
<td></td>
</tr>
<tr>
<td>Internal oscillator</td>
<td>AnalogSubsysRegs.INTOSC1TRIM</td>
<td>Degraded frequency accuracy and temperature drift of the internal oscillators.</td>
</tr>
<tr>
<td></td>
<td>AnalogSubsysRegs.INTOSC2TRIM</td>
<td></td>
</tr>
<tr>
<td>Buffered DAC offset</td>
<td>DacaRegs.DACTRIM</td>
<td>Degraded offset error specification of the buffered DAC. No workaround available.</td>
</tr>
<tr>
<td></td>
<td>DacbRegs.DACTRIM</td>
<td></td>
</tr>
<tr>
<td>PGA gain and offset</td>
<td>PGAGAIN3TRIM</td>
<td>Degraded performance of the PGA gain and offset error specifications. No workaround available.</td>
</tr>
<tr>
<td></td>
<td>PGAGAIN6TRIM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PGAGAIN12TRIM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PGAGAIN24TRIM</td>
<td></td>
</tr>
</tbody>
</table>

**Workaround(s)**  
The following workarounds can be used for improved performance, though it still may not meet data sheet specifications.

Missing **ADC offset trim** can be generated by following the instructions in the ADC Zero Offset Calibration section of the *TMS320F28004x Microcontrollers Technical Reference Manual*.

If the **internal oscillator trim** contains all zeros, the user can adjust the lowest 10 bits of the oscillator trim register between 1 (minimum) and 1023 (maximum) while observing the system clock on the XCLOCKOUT pin.
<table>
<thead>
<tr>
<th>Advisory</th>
<th>PGA: Output Filter Path is Not Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revision(s) Affected</td>
<td>0, A</td>
</tr>
<tr>
<td>Details</td>
<td>The PGA module includes an embedded series-resistor signal path (R_{FILTER}) for implementing a low-pass filter at the PGA_OF pin. This R_{FILTER} signal path should not be used or enabled on the affected revisions. The alternate functions shared with R_{FILTER} on the PGA_OF pin are not affected. For example, ADC input signals A2 and B6 are still available on PGA1_OF.</td>
</tr>
<tr>
<td>Workaround(s)</td>
<td>None</td>
</tr>
<tr>
<td>Advisory</td>
<td>ROM: Flash API Library and FPU32 Twiddle Factor RFFT Table Not Present</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Revision(s) Affected</td>
<td>0, A</td>
</tr>
<tr>
<td>Details</td>
<td>In the affected revisions, the Flash API library and the FPU32 twiddle factor for the 1024-pt RFFT table are not present in the ROM. For details on the ROM contents, see the Memory Maps section of the ROM Code and Peripheral Booting chapter in the TMS320F28004x Microcontrollers Technical Reference Manual.</td>
</tr>
<tr>
<td>Workaround(s)</td>
<td>None</td>
</tr>
</tbody>
</table>
**Advisory**  
**REVID: Some TMX Revision A Devices Have an Incorrect REVID Value**

<table>
<thead>
<tr>
<th>Revision(s) Affected</th>
<th>Details</th>
</tr>
</thead>
</table>
| A                    | Some early TMX Revision A devices have an incorrect value in REVID (address 0x0005_D00C). The REVID incorrectly indicates the Revision 0 value (0x0000_0000) instead of the correct Revision A value (0x0000_0001). Software that uses REVID to distinguish between Revision 0 and Revision A will not function as intended. There are no other functional impacts due to this erratum. Applications that do not use REVID in software will work properly as any other Revision A device. Lot Trace Code affected:  
• 65AVVVDW  
• 66ALSXW |

| Workaround(s) | The device markings on the package are correct and can be used to identify the device revision. |
## Advisory

**GPIO: X2/GPIO18 Pin Pullup Current During Power Up**

<table>
<thead>
<tr>
<th>Revision(s) Affected</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, A, B</td>
<td>During power up, a pullup current of approximately 1.8 mA will be seen on X2/GPIO18. This pin will revert to input mode and operate per the pin description by the time XRSn releases (transitions to high).</td>
</tr>
</tbody>
</table>

| Workaround(s) | None |


Advisory: GPIO: Open-Drain Configuration May Drive a Short High Pulse

Revision(s) Affected
0, A, B

Details
Each GPIO can be configured to an open-drain mode using the GPxODR register. However, an internal device timing issue may cause the GPIO to drive a logic-high for up to 0–10 ns during the transition into or out of the high-impedance state.

This undesired high-level may cause the GPIO to be in contention with another open-drain driver on the line if the other driver is simultaneously driving low. The contention is undesirable because it applies stress to both devices and results in a brief intermediate voltage level on the signal. This intermediate voltage level may be incorrectly interpreted as a high level if there is not sufficient logic-filtering present in the receiver logic to filter this brief pulse.

Workaround(s)
If contention is a concern, do not use the open-drain functionality of the GPIOs; instead, emulate open-drain mode in software. Open-drain emulation can be achieved by setting the GPIO data (GPxDAT) to a static 0 and toggling the GPIO direction bit (GPxDIR) to enable and disable the drive low. For an example implementation, see the code below.

```c
void main(void)
{

    // GPIO configuration
    EALLOW; // disable pullup
    GpioCtrlRegs.GPxPUD.bit.GPIOx = 1; // disable open-drain mode
    GpioCtrlRegs.GPxODR.bit.GPIOx = 0; // set GPIO to drive static 0 before enabling output
    GpioDataRegs.GPxCLEAR.bit.GPIOx = 1;
    EDIS;

    ...  

    // application code
    ...

    // To drive 0, set GPIO direction as output
    GpioCtrlRegs.GPxDIR.bit.GPIOx = 1;

    // To tri-state the GPIO(logic 1), set GPIO as input
    GpioCtrlRegs.GPxDIR.bit.GPIOx = 0;
}
```
Advisory

**GPIO: Parasitic Path to VSS When Maximum V_{IH} is Exceeded in Input Mode**

**Revision(s) Affected**

A

**Details**

If a voltage greater than maximum V_{IH} (V_{DDIO} + 0.3 V) is applied to the GPIO pins listed below, an internal parasitic path from the pin to VSS may be turned on. This parasitic current can impact the functional operation of the pin. This is more likely to occur at high temperature. The parasitic path will be removed when the IO is driven below V_{IL}. The path will not reactivate until another overvoltage event occurs.

- GPIO16
- GPIO17
- GPIO24
- GPIO25
- GPIO26
- GPIO27
- GPIO35 (TDI)
- GPIO37 (TDO)
- GPIO40
- GPIO41
- GPIO42
- GPIO43

Pins configured for output-only mode (with no other drivers on the pin) will not see an overvoltage condition at the pin and are not affected by this advisory.

Pins configured in input or bidirectional mode can see an overvoltage condition in three primary ways:

1. The input is driven by a low-impedance driver that can generate a large overshoot at the input due to impedance mismatch without compensating termination.
2. The input sees large transients from external noise sources that rise above V_{DDIO} + 0.3 V at the pin.
3. The input is driven by a device powered by a different voltage regulator. When receiving voltages from another voltage domain, the system design should always keep voltages below maximum V_{IH}. However, due to the increased possibility of the voltage being temporarily greater than V_{DDIO} + 0.3 V due to a noise event or if there is improper supply sequencing, then this advisory will apply.

**Workaround(s)**

If any of the above conditions apply for an input or bidirectional pin, insert a series resistor between the signal and the input pin. The series resistor should be placed close to the input pin.

If the overvoltage is due to overshoot (situations #1 or #2 above), a series resistor of 100 Ω or greater should be used.

If the overvoltage might be sustained (situation #3 above), a series resistor of 220 Ω or greater should be used.
Advisory  |  GPIO: Pins may Drive High During Power Up
--- | ---
Revision(s) Affected | 0
Details | During power up, the following pins will temporarily be in output mode and drive high. These pins will properly revert to input mode and operate per the pin description by the time XRSn releases (transitions to high).
- GPIO0
- GPIO1
- GPIO2
- GPIO3
- GPIO4
- GPIO5
- GPIO11
- GPIO13
- GPIO29
- GPIO33

Workaround(s) | None
Advisory

GPIO: Signal Latch-up to $V_{SS}$

Revision(s) Affected
0

Details
The ESD structures on the pins listed below can be unintentionally turned on during functional operation, which will pull the pins to $V_{SS}$. There will be approximately 40 mA of additional current on the $V_{DDIO}$ supply for each output pin in this condition.

- GPIO16
- GPIO17
- GPIO24
- GPIO25
- GPIO26
- GPIO27
- GPIO35 (TDI)
- GPIO37 (TDO)
- GPIO40
- GPIO41
- GPIO42
- GPIO43

The condition has not been observed below 70°C under normal operation. This condition can occur in input or output mode and with any of the mux functions. Designs with lightly loaded pins and fast switching signals are more likely to see the condition. Pins not bonded out in smaller pin-count packages can also enter the latch-up condition if they are toggled.

The latch-up condition can be ended by toggling the IO at a lower temperature.

Workaround(s)
Four workaround options are:

1. Avoid using these pins on the revision affected.
2. Avoid high-temperature operations on the revision affected.
3. If the pin is configured as an input or output:
   Place a capacitor of 300 pF or greater between each of these pins and ground, placed as closely as possible to the device. This will slow down the fast signal and avoid triggering the condition. Larger capacitors will be more effective at filtering the transient but must be balanced against the system-level timing requirements of these pins.
   For input pins, a smaller capacitor may be possible when used in combination with option 4.
4. If the pin is configured as an input:
   Connect a resistor in series with any other components on the board such that the total resistance of the driver plus the resistor is 1 kΩ or greater. The goal is to eliminate fast voltage transient seen at the pin. This will also limit the DC current if the ESD structure is activated due to noise.
5 Documentation Support

For more information regarding the F28004x devices, see the following documents:

- TMS320F28004x Microcontrollers Data Manual
- TMS320F28004x Microcontrollers Technical Reference Manual
Trademarks

TMS320 is a trademark of Texas Instruments.
All other trademarks are the property of their respective owners.
## Revision History

Changes from October 23, 2018 to September 13, 2019 (from C Revision (October 2018) to D Revision)                     | Page
---|---
• **Figure 1** (Examples of Device Markings for PM and PZ Packages): Added figure.                           | 2
• **Figure 2** (Example of Device Markings for RSH Package): Added figure.                               | 2
• **Section 4.2** (Known Design Exceptions to Functional Specifications): Changed advisory title from "SDFM: Dynamically Changing Comparator Settings Will Trigger Spurious Comparator Events" to "SDFM: Dynamically Changing Threshold Settings (LLT, HLT), Filter Type, or COSR Settings Will Trigger Spurious Comparator Events". | 12
• **Section 4.2**: Updated SDFM: Dynamically Changing Threshold Settings (LLT, HLT), Filter Type, or COSR Settings Will Trigger Spurious Comparator Events advisory.                           | 12
• **Section 4.2**: Changed advisory title from "SDFM: Dynamically Changing Data Filter Settings Will Trigger Spurious Data Acknowledge Events" to "SDFM: Dynamically Changing Data Filter Settings (Such as Filter Type or DOSR) Will Trigger Spurious Data Acknowledge Events". | 12
• **Section 4.2**: Updated SDFM: Dynamically Changing Data Filter Settings (Such as Filter Type or DOSR) Will Trigger Spurious Data Acknowledge Events advisory.                           | 12
• **Section 4.2**: Added Boot ROM: Calling SCI Bootloader from Application advisory.                        | 25
• **Section 4.2**: Added Memory: Prefetching Beyond Valid Memory advisory.                                 | 26
• **Section 4.2**: Added ADC: Interrupts may Stop if INTxCONT (Continue-to-Interrupt Mode) is not Set advisory. | 27
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
Copyright © 2019, Texas Instruments Incorporated