# TMS320F2802x, TMS320F2802xx Piccolo™ MCUs Silicon Revisions B, A, 0

## 1    Introduction

This document describes the silicon updates to the functional specifications for the TMS320F2802x and TMS320F2802xx microcontrollers (MCUs).

The updates are applicable to:

- 38-pin Thin Shrink Small-Outline Package, DA Suffix
- 48-pin Low-Profile Quad Flatpack, PT 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] DSP devices and support tools. Each TMS320™ DSP commercial family member has one of three prefixes: TMX, TMP, or TMS (for example, **TMS**320F28027). 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, DA) and temperature range (for example, T).

## 3 Device Markings

Figure 1 and Figure 2 provide examples of the 2802x device markings and define each of the markings. The silicon revision can be determined by the symbols marked on the top of the package as shown in Figure 1 and Figure 2 (see Table 1). Some prototype devices may have markings different from those illustrated. Figure 3 and Figure 4 show the device nomenclature.
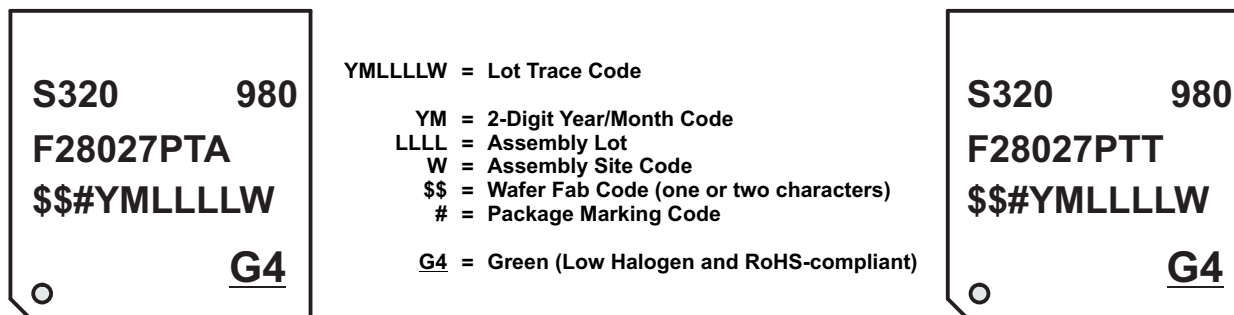
```
S320       980
F28027PTA
$$#YMLLLLW

           G4
○
```

YMLLLLW = Lot Trace Code

YM = 2-Digit Year/Month Code
LLLL = Assembly Lot
W = Assembly Site Code
$$ = Wafer Fab Code (one or two characters)
# = Package Marking Code

G4 = Green (Low Halogen and RoHS-compliant)

```
S320       980
F28027PTT
$$#YMLLLLW

           G4
○
```

**Figure 1. Examples of 2802x Device Markings on PT Package**

```
[TI logo] $$#YMLLLLW
          S320
          F28027DAT

                  G4
```

YMLLLLW = Lot Trace Code

YM = 2-Digit Year/Month Code
LLLL = Assembly Lot
W = Assembly Site Code
$$ = Wafer Fab Code (one or two characters)
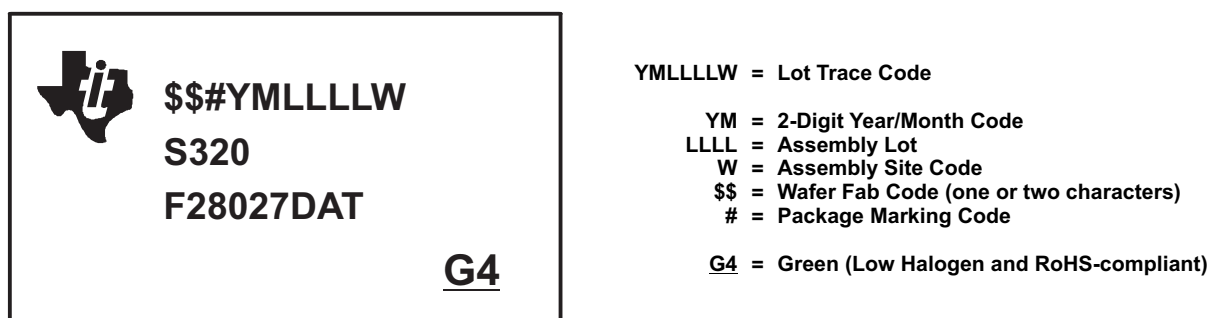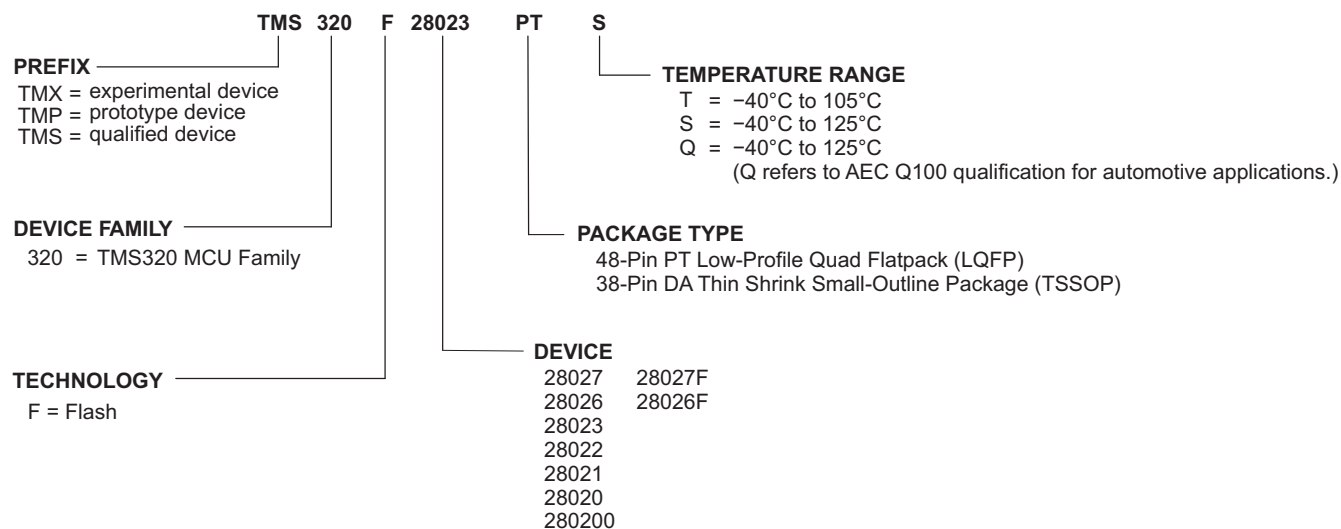# = Package Marking Code

G4 = Green (Low Halogen and RoHS-compliant)

**Figure 2. Example of 2802x Device Markings on DA Package**

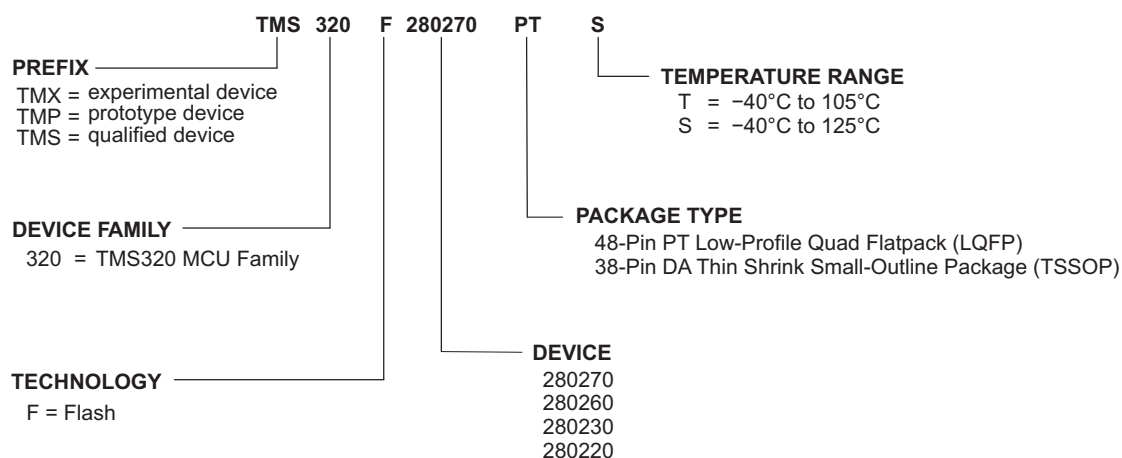**Table 1. Determining Silicon Revision From Lot Trace Code (2802x and 2802xx Devices)[1]**

| PACKAGE MARKING CODE | SILICON REVISION | REVISION ID Address: 0x0883 | COMMENTS |
|---|---|---|---|
| Blank | Indicates Revision 0 | 0x0000 | This silicon revision is available as TMP and TMS. |
| A | Indicates Revision A | 0x0001 | This silicon revision is available as TMS. |
| B | Indicates Revision B | 0x0002 | This silicon revision is available as TMS. |

[1] TMS320F2802xF devices are only available in revision B silicon. All other devices are only available in revision A silicon.

**TMS  320   F  28023   PT   S**

**PREFIX**
TMX = experimental device
TMP = prototype device
TMS = qualified device

**DEVICE FAMILY**
320  = TMS320 MCU Family

**TECHNOLOGY**
F = Flash

**DEVICE**
28027    28027F
28026    28026F
28023
28022
28021
28020
280200

**PACKAGE TYPE**
48-Pin PT Low-Profile Quad Flatpack (LQFP)
38-Pin DA Thin Shrink Small-Outline Package (TSSOP)

**TEMPERATURE RANGE**
T  = −40°C to 105°C
S  = −40°C to 125°C
Q  = −40°C to 125°C
        (Q refers to AEC Q100 qualification for automotive applications.)

A    For more information on peripheral, temperature, and package availability for a specific device, see the
     *TMS320F2802x Piccolo™ Microcontrollers Data Manual*.

**Figure 3. Device Nomenclature for 2802x and 280200**

**TMS  320   F  280270   PT   S**

**PREFIX**
TMX = experimental device
TMP = prototype device
TMS = qualified device

**DEVICE FAMILY**
320  = TMS320 MCU Family

**TECHNOLOGY**
F = Flash

**DEVICE**
280270
280260
280230
280220

**PACKAGE TYPE**
48-Pin PT Low-Profile Quad Flatpack (LQFP)
38-Pin DA Thin Shrink Small-Outline Package (TSSOP)

**TEMPERATURE RANGE**
T  = −40°C to 105°C
S  = −40°C to 125°C

A    For more information on peripheral, temperature, and package availability for a specific device, see the
     *TMS320F2802x0 Piccolo™ Microcontrollers Data Manual*.

**Figure 4. Device Nomenclature for 2802x0**

## 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**

| TITLE | SILICON REVISION(S) AFFECTED | | |
|---|---|---|---|
| | **0** | **A** | **B** |
| PIE: Spurious Nested Interrupt After Back-to-Back PIEACK Write and Manual CPU Interrupt Mask Clear | Yes | Yes | Yes |

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

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

```
        //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
```

## 4.2   Known Design Exceptions to Functional Specifications

### Table 3. Table of Contents for Advisories

Table 4 shows which silicon revision(s) are affected by each advisory.

**Table 4. List of Advisories**

| TITLE | SILICON REVISION(S) AFFECTED | | |
|---|---|---|---|
| | **0** | **A**[1] | **B**[1] |
| ADC: Temperature Sensor Minimum Sample Window Requirement | Yes | Yes | Yes |
| ADC: DC Specifications: Linearity Limitation | Yes | Yes | Yes |
| ADC: Out-of-Specification Current Consumption on Power Up | Yes | Yes | Yes |
| ADC: ADC Result Conversion When Sampling Ends on 14th Cycle of Previous Conversion, ACQPS = 6 or 7 | Yes | Yes | Yes |
| ADC: Initial Conversion | Yes | Yes | Yes |
| ADC: ADC can Become Non-Responsive When ADCNONOVERLAP or RESET is Written During a Conversion | Yes | Yes | Yes |
| ADC: New ADC Control Bits Added to Revision A Silicon | | Yes | Yes |
| BOR: BOR Trip Point Limits $V_{DDIO}$ Tolerance | Yes | Yes | Yes |
| Memory: Prefetching Beyond Valid Memory | Yes | Yes | Yes |
| GPIO: GPIO Qualification | Yes | Yes | Yes |
| Watchdog: Incorrect Operation of CPU Watchdog When WDCLK Source is OSCCLKSRC2 | Yes | Yes | Yes |
| Oscillator: CPU Clock Switching to INTOSC2 May Result in Missing Clock Condition After Reset | Yes | Yes | Yes |
| Flash: Flash API v2.00 Software Library Cannot be Used on the TMS320F28020 and TMS320F280200 Devices | Yes | Yes | Yes |
| Boot ROM: Flash API [Flash_Erase( ) Function] | Yes | Yes | Yes |
| Boot ROM: Flash API [Flash_Program( ) Function] | Yes | | |
| Zero-Pin Oscillator: Modification to Oscillator Frequency Parameter | | Yes | Yes |
| CPU: PCLKCR0[14] Setting May Stall CPU | Yes | | |
| VREG: VREG Minimum $V_{DD}$ Output Voltage for TMP Devices is Below Data Sheet Limit | Yes | | |

(1)    The difference between revision A silicon and revision B silicon is the addition of the InstaSPIN™ code in revision B silicon. The InstaSPIN code in secure ROM is execute-only with no read access. For more details, see the *TMS320F28026F, TMS320F28027F InstaSPIN™-FOC Software Technical Reference Manual*. In revision B silicon, the original ROM contents from revision A silicon have not been modified. All new additions for revision B silicon have been placed in the unused locations of ROM in revision A silicon. However, the ROM revision code has been updated on revision B and the ROM checksum will be different between the two revisions.

| Advisory | ***ADC: Temperature Sensor Minimum Sample Window Requirement*** |
|---|---|

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

**Details**      If the minimum sample window is used (6 ADC clocks at 60 MHz, 116.67 ns), the result of a temperature sensor conversion can have a large error, making it unreliable for the system.

**Workaround(s)**

1. If double-sampling of the temperature sensor is used to avoid the corrupted first sample issue, the temperature sensor result is valid. This is equivalent to giving the S/H circuit adequate time to charge.

2. In all other conditions, the sample-and-hold window used to sample the temperature sensor should not be less than 550 ns.

| Advisory | ***ADC: DC Specifications: Linearity Limitation*** |
|---|---|

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

**Details**      The linearity degrades with increasing temperature in the upper half of the transfer function.

**Workaround(s)**      The impact from this limitation has been addressed in the revision A silicon. The following features have been added:

1. ADC clock divider-by-2 enable bit. At 60 MHz, the effective sample rate will be 2.3 MSPS. This offers a 30-MHz ADC and a 60-MHz system clock, and improves linearity.

2. Existing pipeline mode with 4.6 MSPS at 60-MHz system clock will have improved linearity compared to revision 0 silicon.

> **NOTE:**  For 60-MHz operation, there are periodic missing codes, and INL will be bounded by ±28 LSBs MAX/MIN.
>
> For 30-MHz operation, see the *TMS320F2802x Piccolo™ Microcontrollers Data Manual* and the *TMS320F2802x0 Piccolo™ Microcontrollers Data Manual*.

| **Advisory** | ***ADC: Out-of-Specification Current Consumption on Power Up*** |
|---|---|

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

**Details**   When the ADC module is powered up for the first time, it can consume 2x the current listed ($I_{DDA}$) in the data manual until a sample is initiated.

**Workaround(s)**   Before the ADC is initially powered up, initiate a single conversion. Once the timing of the conversion has been satisfied, an ADC reset is recommended to restore the native state of the ADC before setting up the ADC as done before this addition. (Code is shown below.) If this amount of current can be tolerated by the system, then no change needs to be made to the setup code. Note that the current will go back to within data manual limits once the first conversion is processed by the ADC.

```
EALLOW;
SysCtrlRegs.PCLKCR0.bit.ADCENCLK=1;        //Enable Clocks to ADC module
AdcRegs.ADCSOCFRC1.bit.SOC0 = 1;           //Issue dummy conversion
asm(" rpt #19 || nop");                     //Wait for conversion to propagate
AdcRegs.SOCPRICTL.bit.SOCPRIORITY = 1;     //Change Priority Control to reset
AdcRegs.SOCPRICTL.bit.SOCPRIORITY = 0;     //round robin pointer
EDIS;
```

| **Advisory** | ***ADC: ADC Result Conversion When Sampling Ends on 14th Cycle of Previous Conversion, ACQPS = 6 or 7*** |
|---|---|

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

**Details**   The on-chip ADC takes 13 ADC clock cycles to complete a conversion after the sampling phase has ended. The result is then presented to the CPU on the 14th cycle post-sampling and latched on the 15th cycle into the ADC result registers. If the next conversion's sampling phase terminates on this 14th cycle, the results latched by the CPU into the result register are not assured to be valid across all operating conditions.

**Workaround(s)**   Some workarounds are as follows:

- Due to the nature of the sampling and conversion phases of the ADC, there are only two values of ACQPS (which controls the sampling window) that would result in the above condition occurring—ACQPS = 6 or 7. One solution is to avoid using these values in ACQPS.

- When the ADCNONOVERLAP feature (bit 1 in ADCTRL2 register) is used, the above condition will never be met; so the user is free to use any value of ACQPS desired.

- Depending on the frequency of ADC sampling used in the system, the user can determine if their system will hit the above condition if the system requires the use of ACQPS = 6 or 7. For instance, if the converter is continuously converting with ACQPS = 6, the above condition will never be met because the end of the sampling phase will always fall on the 13th cycle of the current conversion in progress.

| | |
|---|---|
| **Advisory** | ***ADC: Initial Conversion*** |

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

**Details**  When the ADC conversions are initiated by any source of trigger in either sequential or simultaneous sampling mode, the first sample may not be the correct conversion result.

**Workaround(s)**  For sequential mode, discard the first sample at the beginning of every series of conversions. For instance, if the application calls for a given series of conversions, SOC0→SOC1→SOC2, to initiate periodically, then set up the series instead as SOC0→SOC1→SOC2→SOC3 and only use the last three conversions, ADCRESULT1, ADCRESULT2, ADCRESULT3, thereby discarding ADCRESULT0.

For simultaneous sample mode, discard the first sample of both the A and B channels at the beginning of every series of conversions.

User application should validate if this workaround is acceptable in their application.

**The following is applicable to the revision A silicon:**

- For 30-MHz operation and below, this is fixed completely by writing a 1 to the ADCNONOVERLAP and CLKDIV2EN bits in the ADCTRL2 register. This will give a 30-MHz ADC clock when the CPU clock = 60 MHz, and will only allow the sampling of ADC channels when the ADC is finished with any pending conversion.

| | |
|---|---|
| **Advisory** | ***ADC: ADC can Become Non-Responsive When ADCNONOVERLAP or RESET is Written During a Conversion*** |

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

**Details**

The ADC can get into a non-responsive state when the ADCCTL2[ADCNONOVERLAP] is modified while a conversion is in progress. When in this condition, no further conversion from the ADC will be possible without a device reset.

There are two different ways to cause this condition:

- Writing to ADCCTL2[ADCNONOVERLAP] while a conversion is in progress.
- Writing to ADCCTL1[RESET] while a conversion is in progress.

**Workaround(s)**

Follow this sequence to modify ADCCTL2[ADCNONOVERLAP] or write ADCCTL1[RESET]:

1. Set all SOC trigger sources ADCSOCxCTL[TRIGSEL] to 0.
2. Set all ADCINTSOCSEL1/2 to 0.
3. Ensure there is not another SOC pending (This can be accomplished by polling SOC Flags).
4. Wait for all conversions to complete.
   a. ADCCTL2[CLKDIV2EN] = 0, ADCCTL2[CLKDIV4EN] = x $\rightarrow$ (ACQPS + 13 ) * 1 SYSCLKs
   b. ADCCTL2[CLKDIV2EN] = 1, ADCCTL2[CLKDIV4EN] = 0 $\rightarrow$ (ACQPS + 13 ) * 2 SYSCLKs
   c. ADCCTL2[CLKDIV2EN] = 1, ADCCTL2[CLKDIV4EN] = 1 $\rightarrow$ (ACQPS + 13 ) * 4 SYSCLKs
5. Modify ADCCTL2[ADCNONOVERLAP] or write ADCCTL1[RESET].

An example code follows.

```
EALLOW;
// Set all SOC trigger sources to software
AdcRegs.ADCSOC0CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC1CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC2CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC3CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC4CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC5CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC6CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC7CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC8CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC9CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC10CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC11CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC12CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC13CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC14CTL.bit. TRIGSEL  = 0;
AdcRegs.ADCSOC15CTL.bit. TRIGSEL  = 0;

// Set all ADCINTSOCSEL1/2 to 0.
    AdcRegs.ADCINTSOCSEL1.bit.SOC0  = 0;
    AdcRegs.ADCINTSOCSEL1.bit.SOC1  = 0;
    AdcRegs.ADCINTSOCSEL1.bit.SOC2  = 0;
    AdcRegs.ADCINTSOCSEL1.bit.SOC3  = 0;
    AdcRegs.ADCINTSOCSEL1.bit.SOC4  = 0;
    AdcRegs.ADCINTSOCSEL1.bit.SOC5  = 0;
    AdcRegs.ADCINTSOCSEL1.bit.SOC6  = 0;
    AdcRegs.ADCINTSOCSEL1.bit.SOC7  = 0;
    AdcRegs.ADCINTSOCSEL2.bit.SOC8  = 0;
    AdcRegs.ADCINTSOCSEL2.bit.SOC9  = 0;
    AdcRegs.ADCINTSOCSEL2.bit.SOC10 = 0;
    AdcRegs.ADCINTSOCSEL2.bit.SOC11 = 0;
    AdcRegs.ADCINTSOCSEL2.bit.SOC12 = 0;
    AdcRegs.ADCINTSOCSEL2.bit.SOC13 = 0;
    AdcRegs.ADCINTSOCSEL2.bit.SOC14 = 0;
    AdcRegs.ADCINTSOCSEL2.bit.SOC15 = 0;

// Ensure there is not another SOC pending
while (AdcRegs.ADCSOCFLG1.all != 0x0);

// Wait for conversions to complete
// Delay time based on ACQPS = 6 , ADCCTL2[CLKDIV2EN] = 1, ADCCTL2[CLKDIV4EN] = 0
// 7 + 13 ADC Clocks = 20 ADCCLKS -> 40 SYSCLKS
asm(" RPT#40||NOP");
// ADCCTL2[ADCNONOVERLAP] = <new value>;
// ADCCTL1[RESET] = 1;

EDIS;
```

| Advisory | **ADC: New ADC Control Bits Added to Revision A Silicon** |
|---|---|

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

**Details**    The following bits/features have been added to the ADC control registers in the revision A silicon [see the *TMS320x2802x, 2803x Piccolo Analog-to-Digital Converter (ADC) and Comparator Reference Guide* for more information]:

1. Register ADCTL2 at address 0x7101, containing the following bits:
   - CLKDIV2EN, a /2 divide enable that scales CPU Clock by 1/2 for use by the ADC
   - ADCNONOVERLAP, a bit enable that removes the conversion/sample overlap timing
2. Addition of bit field ONESHOT in the existing ADCSOCPRIORITYCTL register. When enabled, the ONESHOT bit field will only let the first SOC complete if multiple SOCs are received at the same time. This is applicable for all SOCs that are in Round-Robin Priority. SOCs in High-Priority mode are not effected by this function.

**Workaround(s)**    Not applicable

| | |
|---|---|
| **Advisory** | ***BOR: BOR Trip Point Limits V$_{DDIO}$ Tolerance*** |

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

**Details**    The maximum V$_{DDIO}$ BOR Trip Point is above the minimum operating voltage of the device. If the full operating range is utilized, random BOR resets may occur.

**Workaround(s)**    Limit the minimum V$_{DDIO}$ operating range to 3.3 V – 5% (3.135 V). Tolerance of ±10% may be used for V$_{DDIO}$ if the BOR is disabled. This can be done right after the device comes out of reset, when the device power consumption is at its minimum. This is done by writing to the BORENZ bit in the BORCFG register. For details on disabling the BOR, see the *TMS320F2802x/TMS320F2802xx Piccolo System Control and Interrupts Reference Guide*.

| | |
|---|---|
| **Advisory** | ***Memory: Prefetching Beyond Valid Memory*** |

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

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

**Workaround**    The prefetch queue is 8 x16 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, OTP, SARAM) on the device. 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.

| | |
|---|---|
| **Advisory** | ***GPIO: GPIO Qualification*** |

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

**Details**    If a GPIO pin is configured for "n" SYSCLKOUT cycle qualification period (where $1 \leq n \leq 510$) 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** | ***Watchdog: Incorrect Operation of CPU Watchdog When WDCLK Source is OSCCLKSRC2*** |
|---|---|
| **Revision(s) Affected** | 0, A, B |
| **Details** | When OSCCLKSRC2 is used as the clock source for CPU watchdog, the watchdog may fail to generate a device reset intermittently. |
| **Workaround(s)** | WDCLK should be sourced only from OSCCLKSRC1 (INTOSC1). The CPU may be sourced from OSCCLKSRC2 or OSCCLKSRC1 (INTOSC1). |

| **Advisory** | ***Oscillator: CPU Clock Switching to INTOSC2 May Result in Missing Clock Condition After Reset*** |
|---|---|
| **Revision(s) Affected** | 0, A, B |
| **Details** | After at least two system resets (not including power-on reset), when the application code attempts to switch the CPU clock source to internal oscillator 2, a missing clock condition will occur, and the clock switching will fail under the following conditions:<br><br>• X1 and X2 are unused (X1 is always tied low when unused).<br>• GPIO38 (muxed with TCK and XCLKIN) is used as JTAG TCK pin only.<br>• JTAG emulator is disconnected.<br><br>The missing clock condition will recover only after a power-on reset when the failure condition occurs. |
| **Workaround(s)** | Before switching the CPU clock source to INTOSC2 via the OSCCLKSRCSEL and OSCCLKSRC2SEL bits in the CLKCTL register, the user must toggle the XCLKINOFF and XTALOSCOFF bits in the CLKCTL register as illustrated in the below sequence:<br><br>```\nCLKCTL |= 0x6000;     // XCLKINOFF = 1, XTALOSCOFF = 1\nCLKCTL &=~0x6000;     // XCLKINOFF = 0, XTALOSCOFF = 0\nCLKCTL |= 0x6000;     // XCLKINOFF = 1, XTALOSCOFF = 1\nCLKCTL &=~0x6000;     // XCLKINOFF = 0, XTALOSCOFF = 0\nCLKCTL |= 0x6000;     // XCLKINOFF = 1, XTALOSCOFF = 1\n```<br><br>Once the above procedure is executed, then the OSC2 selection switches can be configured.<br><br>If the JTAG emulator is connected, and GPIO38 (TCK) is toggling, then the above procedure is unnecessary, but will do no harm.<br><br>If no clock is applied to GPIO38, it is also recommended that a strong pullup resistor on GPIO38 be added to $V_{DDIO}$. |

| **Advisory** | ***Flash: Flash API v2.00 Software Library Cannot be Used on the TMS320F28020 and TMS320F280200 Devices*** |
|---|---|
| **Revision(s) Affected** | 0, A, B |
| **Details** | Version 2.00 of the Flash API is a software-only version (not firmware embedded into the ROM of the device). This library is too large to fit into the RAM of the TMS320F28020 and TMS320F280200 devices. |
| **Workaround(s)** | None for revision 0 silicon. Silicon revision A has v2.01 (functionally equivalent to software v2.00) of the Flash API in the embedded ROM. |

| **Advisory** | ***Boot ROM: Flash API [Flash_Erase( ) Function]*** |
|---|---|

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

**Details**   The Flash API (V1.00) in the device boot ROM contains a coding error in the Flash_Erase( ) function. This issue does not affect the flash erase operation. On an Erase failure, the function would fail and return error status as expected, but the error address field does not get updated correctly.

**Workaround(s)**   None

| **Advisory** | ***Boot ROM: Flash API [Flash_Program( ) Function]*** |
|---|---|

**Revision(s) Affected**   0

**Details**   The Flash API (V1.00) in the device boot ROM contains a coding error in the Flash_Program( ) function.

**Workaround(s)**   **For the TMS320F28027, TMS320F28023, or TMS320F28021 devices:** Follow the instructions in the Flash2802x_API_Readme.pdf file of the Flash API V1.00 release to replace the Flash_Program( ) function in the boot ROM with the version in the Flash API V1.00a software library. Alternatively, use the Flash API V2.00 software library.

**For the TMS320F28026 or TMS320F28022 devices:** Use the Flash API V2.00 software library.

The Boot ROM in Silicon Revision A will be updated to the Flash API V2.01 to fix this issue. The ROM symbol library will be updated to V2.01, which will be functionally equivalent to the V2.00 software library. Any application which uses the Flash API symbol library V1.00 will need to be rebuilt using the updated symbol library. Any application which uses the V2.00 software library can optionally upgrade to the V2.01 symbol library to conserve RAM memory. This has been fixed in revision A silicon.

| Advisory | *Zero-Pin Oscillator: Modification to Oscillator Frequency Parameter* |
|---|---|
| **Revision(s) Affected** | A, B |
| **Details** | The zero-pin oscillator is now specified with the center frequency at a defined temperature and temperature coefficient to calculate the absolute frequency at any operational temperature. Customers will need to check their temperature profile to ensure the zero-pin oscillator meets their requirement. |
| **Workaround(s)** | If the frequency output does not meet a needed tolerance, software compensation can be used to achieve better than 1% frequency spread about 10 MHz at any temperature. |

| Advisory | *CPU: PCLKCR0[14] Setting May Stall CPU* |
|---|---|
| **Revision(s) Affected** | 0 |
| **Details** | Setting Bit 14 of the PCLKCR0 register (PCLKCR0[14]) to "1" may stall the CPU and inadvertently secure the device upon subsequent operations. |
| **Workaround(s)** | Do not set the PCLKCR0[14] bit to "1" in revision 0 silicon. This has been fixed in revision A silicon. |

| Advisory | *VREG: VREG Minimum $V_{DD}$ Output Voltage for TMP Devices is Below Data Sheet Limit* |
|---|---|
| **Revision(s) Affected** | 0 |
| **Details** | The minimum $V_{DD}$ output voltage when the VREG is enabled is 1.8 V for TMP-marked devices. This is below the data sheet limit of 1.865 V. |
| **Workaround(s)** | None. This has been fixed in the TMS devices (all revisions). |

## 5    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 Piccolo devices, see the following documents:

- *TMS320F2802x Piccolo™ Microcontrollers Data Manual*
- *TMS320F2802x0 Piccolo™ Microcontrollers Data Manual*

## Trademarks

Piccolo, TMS320, InstaSPIN are trademarks of Texas Instruments.
All other trademarks are the property of their respective owners.

# Revision History

**Changes from January 3, 2018 to April 4, 2018 (from N Revision (January 2018) to O Revision)**      **Page**