

F29H85x and F29P58x Real-Time MCUs Silicon Errata (Silicon Revision 0)



ABSTRACT

This document describes the known exceptions to the functional specifications (advisories). This document may also contain usage notes. Usage notes describe situations where the device's behavior may not match presumed or documented behavior. This may include behaviors that affect device performance or functional correctness.

Table of Contents

1 Usage Notes and Advisories Matrices	2
1.1 Usage Notes Matrix.....	2
1.2 Advisories Matrix.....	2
2 Nomenclature, Package Symbolization, and Revision Identification	3
2.1 Device and Development-Support Tool Nomenclature.....	3
2.2 Devices Supported.....	3
2.3 Package Symbolization and Revision Identification.....	4
3 Silicon Revision 0 Usage Notes and Advisories	6
3.1 Silicon Revision 0 Usage Notes.....	6
3.2 Silicon Revision 0 Advisories.....	7
4 Documentation Support	20
5 Trademarks	20
6 Revision History	20

List of Figures

Figure 2-1. Package Symbolization for ZEX Package.....	4
Figure 2-2. Package Symbolization for PTS Package.....	4
Figure 2-3. Package Symbolization for RFS Package.....	4
Figure 2-4. Package Symbolization for PZS Package.....	5
Figure 3-1. Undesired Trip Event and Blanking Window Expiration.....	10
Figure 3-2. Resulting Undesired ePWM Outputs Possible.....	10
Figure 3-3. Incorrect Power-up Sequence Leading to Stuck Reset.....	17
Figure 3-4. Correct Power-up Sequence With Reset Release.....	17

List of Tables

Table 1-1. Usage Notes Matrix.....	2
Table 1-2. Advisories Matrix.....	2
Table 2-1. Revision Identification.....	5

1 Usage Notes and Advisories Matrices

Table 1-1 lists all usage notes and the applicable silicon revisions. Table 1-2 lists all advisories, modules affected, and the applicable silicon revisions.

1.1 Usage Notes Matrix

Table 1-1. Usage Notes Matrix

NUMBER	TITLE	SILICON REVISIONS AFFECTED
		0
TBD	TBD	Yes

1.2 Advisories Matrix

Table 1-2. Advisories Matrix

MODULE	DESCRIPTION	SILICON REVISIONS AFFECTED
		0
ADC	ADC: Interrupts may Stop if INTxCONT (Continue-to-Interrupt Mode) is not Set	Yes
DCAN	During DCAN FIFO Mode, Received Messages May be Placed Out of Order in the FIFO Buffer	Yes
MCAN	Message Order Inversion When Transmitting From Dedicated Tx Buffers Configured With Same Message ID	Yes
ePWM	ePWM: An ePWM Glitch can Occur if a Trip Remains Active at the End of the Blanking Window	Yes
Flash	Flash: Stand-alone CPU1/CPU3 Reset With Flash Prefetch Enabled May Cause NMI to CPU1/CPU3	Yes
GPIO	GPIO: Open-Drain Configuration may Drive a Short High Pulse	Yes
MCD	MCD: Missing Clock Detect Should be Disabled When the PLL is Enabled (PLLCLKEN = 1)	Yes
SDFM	SDFM: Dynamically Changing Threshold Settings (LLT, HLT), Filter Type, or COSR Settings Will Trigger Spurious Comparator Events	Yes
SDFM	SDFM: Dynamically Changing Data Filter Settings (Such as Filter Type or DOSR) Will Trigger Spurious Data Acknowledge Events	Yes
SDFM	SDFM: Two Back-to-Back Writes to SDCPARMx Register Bit Fields CEVT1SEL, CEVT2SEL, and HZEN Within Three SD-Modulator Clock Cycles can Corrupt SDFM State Machine, Resulting in Spurious Comparator Events	Yes
C29 CPU Subsystem	C29 CPU Subsystem: DTHE Interrupts and DMA Events Not Triggered in C29 CPU Subsystem for HS-FS Devices	Yes
System	System: Device Reset Remains Asserted When VDD Voltage Ramps Before VDDIO	Yes
System	System: Pending Misaligned Reads in the Pipeline After CPU Goes to Fault State Preventing NMI Vector Fetch	Yes
UART	UART: UART FIFO Gets Cleared on Continuous Debugger Reads	Yes

2 Nomenclature, Package Symbolization, and Revision Identification

2.1 Device and Development-Support Tool Nomenclature

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 (TMDX) through fully qualified production tools (TMDS).

Device development evolutionary flow:

- X** Experimental device that is not necessarily representative of the final device's electrical specifications and may not use production assembly flow.
- P** Prototype device that is not necessarily the final silicon die and may not necessarily meet final electrical specifications.
- null** Production version of the silicon die that is fully qualified.

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.

X and P devices and TMDX development-support tools are shipped against the following disclaimer:

"Developmental product is intended for internal evaluation purposes."

Production 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 (X or P) 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.


2.2 Devices Supported

This document supports the following devices:

- [F29H850TU](#)
- [F29H859TU-Q1](#)
- F29H859TM-Q1
- F29H850DU
- F29H859DU-Q1
- F29H850DM
- F29H859DM-Q1
- F29P589DU-Q1
- F29P580DM
- F29P589DM-Q1

2.3 Package Symbolization and Revision Identification

Figure 2-1, Figure 2-2, Figure 2-3, and Figure 2-4 show the package symbolization. Table 2-1 lists the silicon revision codes.




YMLLLLLS = Lot Trace Code

- \$\$ = Wafer Fab Code (one or two characters) as applicable**
- # = Silicon Revision Code**
- YM = 2-digit Year/Month Code**
- LLLL = Assembly Lot Code**
- S = Assembly Site Code**

G1 = Green (Low Halogen and RoHS-compliant)

Package Pin 1

Figure 2-1. Package Symbolization for ZEX Package



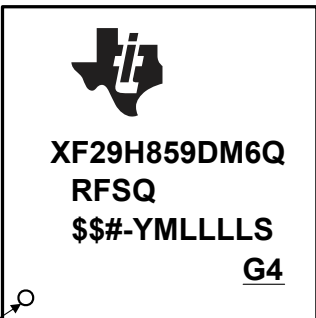
YMLLLLLS = Lot Trace Code

- \$\$ = Wafer Fab Code (one or two characters) as applicable**
- # = Silicon Revision Code**
- YM = 2-digit Year/Month Code**
- LLLL = Assembly Lot Code**
- S = Assembly Site Code**

G4 = Green (Low Halogen and RoHS-compliant)

Package Pin 1

Figure 2-2. Package Symbolization for PTS Package



YMLLLLLS = Lot Trace Code

- \$\$ = Wafer Fab Code (one or two characters) as applicable**
- # = Silicon Revision Code**
- YM = 2-digit Year/Month Code**
- LLLL = Assembly Lot Code**
- S = Assembly Site Code**

G4 = Green (Low Halogen and RoHS-compliant)

Package Pin 1

Figure 2-3. Package Symbolization for RFS Package

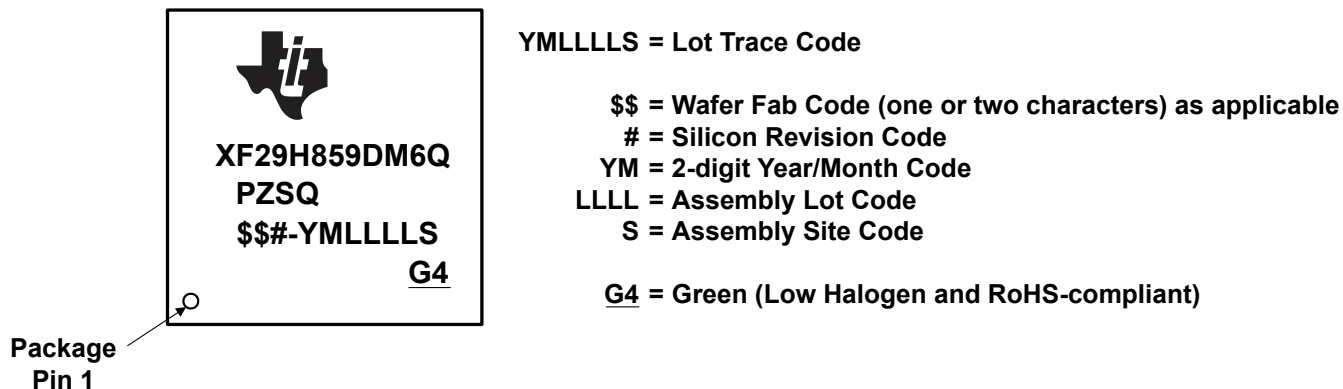


Figure 2-4. Package Symbolization for PZS Package

Table 2-1. Revision Identification

SILICON REVISION CODE	SILICON REVISION	REVID ⁽¹⁾ Address: 0x5D00C	COMMENTS ⁽²⁾
Blank	0	0x0000 0000	This silicon revision is available as pre-production.

- (1) Silicon Revision ID
- (2) For orderable device numbers, see the PACKAGING INFORMATION table in the [F29H85x and F29P58x Real-Time Microcontrollers](#) data sheet.

3 Silicon Revision 0 Usage Notes and Advisories

This section lists the usage notes and advisories for this silicon revision.

3.1 Silicon Revision 0 Usage Notes

This section lists all the usage notes that are applicable to silicon revision 0.

3.2 Silicon Revision 0 Advisories

This section lists all the advisories that are applicable to silicon revision 0.

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

Revisions Affected 0

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.

Workarounds

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 ***During DCAN FIFO Mode, Received Messages May be Placed Out of Order in the FIFO Buffer***

Revisions Affected 0

Details

In DCAN FIFO mode, received messages with the same arbitration and mask IDs are supposed to be placed in the FIFO in the order in which they are received. The CPU then retrieves the received messages from the FIFO via the IF1/IF2 interface registers. Some messages may be placed in the FIFO out of the order in which they were received. If the order of the messages is critical to the application for processing, then this behavior will prevent the proper use of the DCAN FIFO mode.

Workaround

Use the DMA to read out the FIFO via the IF3 register. Each time a message is received into the FIFO, the data is also copied to the IF3 register, and a DMA request is generated to the DMA module to read out the data.

Advisory ***Message Order Inversion When Transmitting From Dedicated Tx Buffers
Configured With Same Message ID***

Revisions Affected 0

Details

Multiple Tx Buffers are configured with the same Message ID. Transmission of these Tx buffers is requested sequentially in ascending order with a delay between the individual Tx requests. Depending on the delay between the individual Tx requests, the Tx Buffers may not be transmitted in the expected ascending order of the Tx Buffer number.

Workarounds

First, write the group of Tx messages with same Message ID to the Message RAM. Then, request transmission of all of these messages concurrently by a single write access to **TXBAR**.

Use the Tx FIFO instead of dedicated Tx Buffers for the transmission of several messages with the same Message ID in a specific order.

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

Revisions Affected 0

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 3-1 illustrates the time period which could result in an undesired ePWM output.

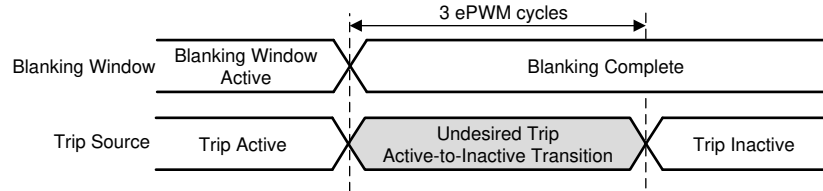


Figure 3-1. Undesired Trip Event and Blanking Window Expiration

Figure 3-2 illustrates the two potential ePWM outputs possible if the trip event ends within 1 cycle before or 3 cycles after the blanking window closes.

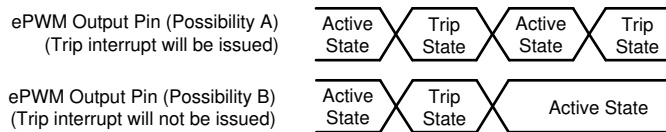


Figure 3-2. Resulting Undesired ePWM Outputs Possible

Workaround

Extend or reduce the blanking window to avoid any undesired trip action.

Advisory *Flash: Stand-alone CPU1/CPU3 Reset With Flash Prefetch Enabled May Cause NMI to CPU1/CPU3*

Revisions Affected 0

Details

If Flash prefetch is enabled, a stand-alone reset issued to CPU1 or CPU3 (for example, debug reset from CCS) may cause an NMI to the CPU because of an uncorrectable ECC error. Below are the sources for a stand-alone CPU reset.

CPU1:

1. Debug reset
2. HSM → CPU1.RSn

CPU3:

1. Debug reset
2. HSM → CPU3.RSn
3. CPU3 WD Reset
4. CPU3 NMIWD Reset
5. SSU → CPU_RST_CTRL.SW_SYSRN

Workaround

Disable Flash prefetch (FRIx_INTF_CTRL.PREFETCH_EN = 0) before issuing a stand-alone reset to the CPU.

CPU1 → FRI1_INTF_CTRL.PREFETCH_EN = 0

CPU3 → FRI3_INTF_CTRL.PREFETCH_EN = 0

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

Revisions Affected 0

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

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.

```

void main(void)
{ ...

    // GPIO configuration
    EALLOW;
    GpioCtrlRegs.GPxPUD.bit.GPIOx = 1;    // disable pullup
    GpioCtrlRegs.GPxODR.bit.GPIOx = 0;    // disable open-drain mode
                                           // 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 ***MCD: Missing Clock Detect Should be Disabled When the PLL is Enabled
(PLLCLKEN = 1)***

Revisions Affected 0

Details

The PLL has a limp mode feature to provide a slow PLLRAWCLK output even if its input OSCCLK is absent. Independently, the Missing Clock Detect (MCD) circuit will forcibly switch the system clock source to INTOSC1 when a missing OSCCLK input is detected. The MCD mux to switch between these system clock sources is not ensured to be glitch-free when both clock sources (PLLRAWCLK and INTOSC1) are still active. In rare cases, this may lead to unpredictable device behavior during a missing clock failure event.

Workarounds

When the PLL is used by the system (PLLCLKEN = 1), disable the MCD by writing MCDCR.MCLKOFF = 1.

The Dual Clock Comparator (DCC) circuit can be configured to quickly detect if the SYSCLK frequency drops outside the desired frequency to its limp mode due to a missing clock event.

When the system is operating in PLL bypass mode (PLLCLKEN = 0), the MCD circuit can still be used to detect missing clock events and switch the clock source to INTOSC1.

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

Revisions Affected 0
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, ePWM X-BAR events, and GPIO output X-BAR events if configured appropriately.

Workaround

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

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

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

Revisions Affected 0
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 and DMA trigger if configured appropriately.

Workaround

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

Advisory ***SDFM: Two Back-to-Back Writes to SDCPARMx Register Bit Fields CEVT1SEL, CEVT2SEL, and HZEN Within Three SD-Modulator Clock Cycles can Corrupt SDFM State Machine, Resulting in Spurious Comparator Events***

Revisions Affected 0

Details Back-to-back writes to SDCPARMx register bit fields CEVT1SEL, CEVT2SEL, and HZEN within three SD-modulator clock cycles can potentially corrupt the SDFM state machine, resulting in spurious comparator events, which can potentially trigger CPU interrupts, ePWM XBAR events, and GPIO output X-BAR events if configured appropriately.

Workaround Avoid back-to-back writes within three SD-modulator clock cycles or have the SDCPARMx register bit fields configured in one register write.

Advisory ***C29 CPU Subsystem: DTBE Interrupts and DMA Events Not Triggered in C29 CPU Subsystem for HS-FS Devices***

Revisions Affected 0**Details**

In the High-Security, Field-Securable (HS-FS) device life-cycle state, cryptographic engines can be automatically assigned to the C29 CPU subsystem using a boot certificate extension option. When this option is configured in the certificate, the engines are mapped to the C29 CPU, but the corresponding interrupt signals and DMA events are not routed to the C29 CPU, and thus do not trigger when running a C29 application.

Workaround

To detect events generated by the cryptographic accelerator engines, poll the respective interrupt or DMA status register.

Advisory **System: Device Reset Remains Asserted When VDD Voltage Ramps Before VDDIO**

Revisions Affected 0

Details

The device XRSn reset signal can remain in a low (reset asserted) state when the VDD supply voltage is ramped up before or simultaneously with the VDDIO supply. As a result, the device fails to boot, and draws a large current on the VDD supply.

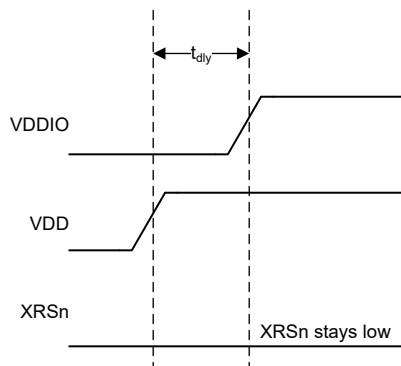


Figure 3-3. Incorrect Power-up Sequence Leading to Stuck Reset

Workaround

Ensure the VDDIO supply is fully ramped up at least 1ms before ramping up the VDD supply voltage.

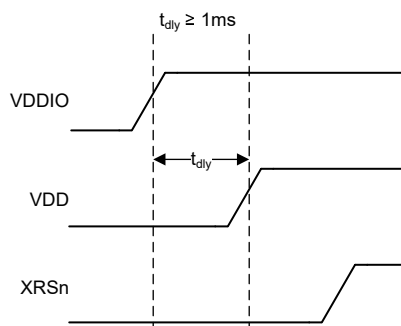


Figure 3-4. Correct Power-up Sequence With Reset Release

Advisory **System: Pending Misaligned Reads in the Pipeline After CPU Goes to Fault State Preventing NMI Vector Fetch**

Revisions Affected 0**Details**
The NMI handler fails to execute when three or more back-to-back C29 CPU faults caused by misaligned reads occur. When more than two faults are in the CPU pipeline, the CPU does not fetch the NMI vector as expected.**Workaround**
Use ERAD-SEC counter:

1. Choose ESM_GEN_EVENT as input to EPWMXBAR.
2. Configure the ERAD-SEC1 counter in start-stop mode: start event as EPWMXBAR event, stop event as SEC1 event itself. This counter will be counting SYSCLK cycles.
3. Configure the ERAD-SEC reference register to generate a match event and trigger an NMI (INT_EN and NMI_EN bits in SEC_CNTL register) on a count of 50.
4. Configure ESM CPU1 to generate an NMI on the ERAD_CPU1_NMI event.

Advisory ***UART: UART FIFO Gets Cleared on Continuous Debugger Reads***

Revisions Affected 0

Details The UART IP treats debugger and CPU reads in the same way. As a result, on reading the UART_DR register continuously from CCS, the FIFO gets cleared before the code actually reads it.

Workaround Do not keep the memory browser open during UART data transfers.

4 Documentation Support

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

For more information regarding the F29H85x and F29P58x devices, see the following documents:

- [F29H85x and F29P58x Real-Time Microcontrollers](#) data sheet
- [F29H85x and F29P58x Real-Time Microcontrollers Technical Reference Manual](#)

5 Trademarks

All trademarks are the property of their respective owners.

6 Revision History

DATE	REVISION	NOTES
November 2024	*	<p>Initial Release</p> <p>TI is transitioning to use more inclusive terminology. Some language may be different than what you would expect to see for certain technology areas.</p> <p>For SPI, all instances of legacy terminology have been changed to controller and peripheral. All instances of legacy pin names have been changed to: POCI (Peripheral OUT Controller IN); PICO (Peripheral IN Controller OUT); and CS (Chip Select).</p> <p>For the I2C Bus Interface, all instances of legacy terminology have been changed to controller and target.</p> <p>For the CAN and LIN Interface/BUS, all instances of legacy terminology have been changed to commander and responder.</p> <p>For the EtherCAT Controller, all instances of legacy terminology have been changed to MainDevice (or MDevice) and SubordinateDevice (or SubDevice).</p>

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), 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, regulatory 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](#) or other applicable terms available either on [ti.com](https://www.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.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2024, Texas Instruments Incorporated