

MSP430F21x2 Device Erratasheet

1 Current Version

See [Appendix A](#) for prior silicon versions.



✓ The checkmark means that the issue is present in the specified revision.

| Devices | Rev: | BCL12 | CPU19 | FLASH19 | FLASH24 | FLASH27 | PORT12 | TA12 | TA16 | TA22 | USCI20 | USCI21 | USCI22 | USCI23 | USCI24 | USCI25 | USCI26 | USCI28 | XOSC5 | XOSC8 |
|-------------|------|-------|-------|---------|---------|---------|--------|------|------|------|--------|--------|--------|--------|--------|--------|--------|--------|-------|-------|
| MSP430F2112 | B | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| MSP430F2122 | B | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| MSP430F2132 | B | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

2 Package Markings

PW28

TSSOP (PW), 28 Pin

| | |
|---|---|
| <div> 4Fxxxxx  YMS # ○ LLLL </div> | YM = Year and Month Date Code LLLL = LOT Trace Code S = Assembly Site Code # = DIE Revision o = PIN 1 |
| <div> MSP430Fxxxx  YMS <u>G</u>4 ○ LLLL # </div> | YM = Year and Month Date Code LLLL = LOT Trace Code S = Assembly Site Code # = DIE Revision o = PIN 1 |

RHB32

QFN (RHB), 32 Pin

| | |
|---|---|
| ○ MSP430 Fxxxx TI YMS# LLLL <u>G</u> 4 | YM = Year and Month Date Code LLLL = LOT Trace Code S = Assembly Site Code # = DIE Revision o = PIN 1 |
|---|---|

3 Detailed Bug Description

BCL12 *Basic Clock Module*

Function Switching RSEL can cause DCO dead time

Description After switching RSELx bits (located in register BCCTL1) from a value of >13 to a value of <12 OR from a value of <12 to a value of >13, the resulting clock delivered by the DCO can stop before the new clock frequency is applied. This dead time is approximately 20 μ s. In some instances, the DCO may completely stop, requiring a power cycle.

Workaround

- When switching RSEL from >13 to <12, use an intermediate frequency step. The intermediate RSEL value should be 13.

| CURRENT RSEL | TARGET RSEL | RECOMMENDED TRANSITION SEQUENCE |
|--------------|-------------|---|
| 15 | 14 | Switch directly to target RSEL |
| 14 or 15 | 13 | Switch directly to target RSEL |
| 14 or 15 | 0 to 12 | Switch to 13 first, and then to target RSEL (two step sequence) |
| 0 to 13 | 0 to 12 | Switch directly to target RSEL |

- When switching RSEL from <12 to >13, ensure that the maximum system frequency is not exceeded during the transition. This can be achieved by clearing the DCO bits first (DCOCTL control register, bits 7–5), then increasing the RSEL value, and finally applying the target frequency DCO bit values. For more details, see the examples in the "TLV Structure" chapter in the *MSP430x2xx Family User's Guide* ([SLAU144](#)).

CPU19 *CPU Module*

Function CPUOFF can change register values

Description If a CPUOFF command is followed by an instruction with an indirect addressed operand (for example, `mov @R8, R9, and RET`), an unintentional register-read operation occur during the wakeup of the CPU. If the unintentional read occurs to a read-sensitive register (for example, UCB0RXBUF or TAIV), which changes its value or the value of other registers (IFGs), the bug leads to lost interrupts or wrong register read values.

Workaround Insert a NOP instruction after each CPUOFF instruction.

FLASH19 *Flash Module*

Function EEI feature does not work for code execution from RAM

Description When the program is executed from RAM, the flash controller EEI feature does not work. The erase cycle is suspended, and the interrupt is serviced, but there is a problem while resuming with the erase cycle.

Addresses applied to flash are different from the actual values while resuming erase cycle after ISR execution.

Workaround None

FLASH24

Flash Module

Function

Write or erase emergency exit can cause failures

Description

When a flash write or erase is abruptly terminated, the following flash accesses by the CPU may be unreliable and result in erroneous code execution. The abrupt termination can be the result of one the following events:

1. The Flash Controller Clock is configured to be sourced by an external crystal. An oscillator fault occurs thus stopping this clock abruptly.
or
2. The Emergency Exit bit (EMEX in FCTL3) when set forces a write or an erase operation to be terminated before normal completion.
or
3. The Enable Emergency Interrupt Exit bit (EEIEX in FCTL1) when set with GIE = 1 can lead to an interrupt causing an emergency exit during a Flash operation.

Workaround

1. Use the internal DCO as the flash controller clock provided from MCLK or SMCLK.
or
2. After setting EMEX = 1, wait for a sufficient amount of time before flash is accessed again.
or
3. No workaround. Do not use EEIEX bit.

FLASH27

Flash Module

Function

EEL feature can disrupt segment erase

Description

When a flash segment erase operation is active with EEI feature selected (EEI = 1 in FLCTL1) and GIE = 0, the following can occur:

An interrupt event causes the flash erase to be stopped, and the flash controller expects an RETI to resume the erase. Because GIE = 0, interrupts are not serviced and RETI never happens.

Workaround

- Do not set bit EEI = 1 when GIE = 0.
or
- Force an RETI instruction during the erase operation during the check for BUSY=1 (FCTL3).

Sample Code:

```

MOV    R5, 0(R5)           ; Dummy write, erase segment
LOOP:  BIT    #BUSY, &FCTL3 ; test busy bit
        JMP    SUB_RETI      ; Force RETI instruction
        JNZ    LOOP          ; loop while BUSY=1

SUB_RETI:  PUSH  SR
          RETI

```

| | |
|--------------------|---|
| PORT12 | <i>Digital I/O Module, Port 1 and 2</i> |
| Function | PxIFG is set on PUC |
| Description | The PxIN register is cleared when a PUC is asserted, and it regains the original value after the PUC is de-asserted. If the PxIN register bits read high, asserting a PUC causes clearing of the register, which results in a high-to-low transition. Once the PUC is de-asserted, the PxIN register is restored to high, which results in a low-to-high transition. This behavior results in the PxIFG being set regardless of the PxIES setting. |
| Workaround | Prior to setting PxIE bits, ensure that corresponding PxIFG bits are cleared. |
| TA12 | <i>Timer_A Module</i> |
| Function | Interrupt is lost (slow ACLK) |
| Description | <p>Timer_A counter is running with slow clock (external TACLK or ACLK) compared to MCLK. The compare mode is selected for the capture/compare channel and the CCRx register is incremented by one with the occurring compare interrupt (if TAR = CCRx).</p> <p>Due to the fast MCLK, the CCRx register increment (CCRx = CCRx + 1) happens before the Timer_A counter has incremented again. Therefore, the next compare interrupt should happen at once with the next Timer_A counter increment (if TAR = CCRx + 1). This interrupt is lost.</p> |
| Workaround | Switch capture/compare mode to capture mode before the CCRx register increment. Switch back to compare mode afterward. |
| TA16 | <i>Timer_A Module</i> |
| Function | First increment of TAR erroneous when IDx > 00 |
| Description | The first increment of TAR after any timer clear event (POR/TACLR) happens immediately following the first positive edge of the selected clock source (INCLK, SMCLK, ACLK, or TACLK). This is independent of the clock input divider settings (ID0, ID1). All following TAR increments are performed correctly with the selected IDx settings. |
| Workaround | None |
| TA22 | <i>Timer_A Module</i> |
| Function | Timer_A register modification after watchdog timer PUC |
| Description | Unwanted modification of the Timer_A registers TACTL and TAIV can occur when a PUC is generated by the watchdog timer (WDT) in watchdog mode and any Timer_A counter register TACCRx is incremented/decremented (Timer_A does not need to be running). |
| Workaround | Initialize TACTL register after the reset occurs using a MOV instruction (BIS or BIC may not fully initialize the register). TAIV is automatically cleared following this initialization. |
| Example | <pre>MOV.W #VAL, &TACTL</pre> <p>Where VAL = 0, if Timer is not used in application; otherwise, user defined per desired function.</p> |

| USCI20 | <i>USCI Module</i> |
|--------------------|---|
| Function | I ² C mode multi-master transmitter issue |
| Description | <p>When configured for I²C master-transmitter mode and used in a multi-master environment, the USCI module can cause unpredictable bus behavior if all of the following conditions are true:</p> <ol style="list-style-type: none"> 1. Two masters are generating SCL. and 2. The slave is stretching the SCL low phase of an ACK period while outputting NACK on SDA. and 3. The slave drives ACK on SDA after the USCI has already released SCL, and then the SCL bus line is released. and 4. The transmit buffer has not been loaded before the other master continues communication by driving SCL low. <p>The USCI remains in the SCL high phase until the transmit buffer is written. After the transmit buffer has been written, the USCI interferes with the current bus activity and may cause unpredictable bus behavior.</p> |
| Workaround | <ul style="list-style-type: none"> • Ensure that slave does not stretch the SCL low phase of an ACK period. or • Ensure that the transmit buffer is loaded in time. or • Do not use the multi-master transmitter mode. |

USCI21
USCI Module
Function

UART IrDA receiver filter

Description

The IrDA receive filter can be used to filter pulses with length UCAIRRXFL configured in UCAXIRRCTL register. If UCIRRXFE is set the IrDA receive decoder may filter out pulses longer than the configured filter length depending on frequency of BRCLK. This results in framing errors or corrupted data on the receiver side.

Workaround

Depending on the used baud rate and the configured filter length, a maximum frequency for BRCLK needs to be set to avoid this issue:

$$\text{Max BRCLK} = \frac{\text{Filter Length} + 64}{2} \times \frac{\text{Baud Rate} \times 16}{3 \times 10^6}$$

| Baud Rate | Filter Length UCIRRXFL (dec) | Max BRCLK (MHz) |
|-----------|---------------------------------|-----------------|
| 9600 | 64 | 3.28 |
| | 32 | 2.46 |
| | 16 | 2.05 |
| | 8 | 1.84 |
| | 4 | 1.74 |
| | 2 | 1.69 |
| | 1 | 1.66 |
| | 0 | 1.64 |
| 19200 | 64 | 6.55 |
| | 32 | 4.92 |
| | 16 | 4.1 |
| | 8 | 3.69 |
| | 4 | 3.48 |
| | 2 | 3.38 |
| | 1 | 3.33 |
| | 0 | 3.28 |
| 38400 | 64 | 13.11 |
| | 32 | 9.83 |
| | 16 | 8.19 |
| | 8 | 7.37 |
| | 4 | 6.96 |
| | 2 | 6.76 |
| | 1 | 6.66 |
| | 0 | 6.55 |
| 56000 | 64 | 19.11 |
| | 32 | 14.34 |
| | 16 | 11.95 |
| | 8 | 10.75 |
| | 4 | 10.15 |
| | 2 | 9.86 |
| | 1 | 9.71 |
| | 0 | 9.56 |

| | |
|--------------------|--|
| USCI22 | <i>USCI Module</i> |
| Function | I ² C master receiver with 10-bit slave addressing |
| Description | <p>Unexpected behavior of the USCI_B can occur when configured in I²C master receive mode with 10-bit slave addressing under the following conditions:</p> <ol style="list-style-type: none"> 1. The USCI sends first byte of slave address, the slave sends an ACK and when second address byte is sent, the slave sends a NACK. 2. Master sends a repeat start condition (if UCTXSTT = 1). 3. The first address byte following the repeated start is acknowledged. <p>However, the second address byte is not sent; instead, the master incorrectly starts to receive data and sets UCBxRXIFG = 1.</p> |
| Workaround | Do not use a repeated start condition; instead, set the stop condition UCTXSTP = 1 in the NACK ISR prior to the following start condition (USTXSTT = 1). |
| USCI23 | <i>USCI Module</i> |
| Function | UART transmit mode with automatic baud rate detection |
| Description | Erroneous behavior of the USCI_A can occur when configured in UART transmit mode with automatic baud rate detection. During transmission if a "Transmit break" is initiated (UCTXBRK = 1), the USCI_A does not deliver a stop bit of logic high; instead, it sends a logic low during the subsequent synch period. |
| Workaround | <ul style="list-style-type: none"> • Follow user's guide instructions for transmitting a break/synch field following UCSWRST = 1. or • Set UCTXBRK = 1 before an active transmission; that is, check for bit UCBUSY = 0 and then set UCTXBRK = 1. |
| USCI24 | <i>USCI Module</i> |
| Function | Incorrect baud rate information during UART automatic baud rate detection mode |
| Description | Erroneous behavior of the USCI_A can occur when configured in UART mode with automatic baud rate detection. After automatic baud rate measurement is complete, the UART updates UCAxBR0 and UCAxBR1. Under oversampling mode (UCOS16 = 1), for baud rates that should result in UCAxBRx = 0x0002, the UART incorrectly reports it as UCAxBRx = 0x5555. |
| Workaround | When break/synch is detected following the automatic baud rate detection, the flag UCBRK flag is set to 1. Check if UCAxBRx = 0x5555 and correct it to 0x0002. |
| USCI25 | <i>USCI Module</i> |
| Function | TXIFG is not reset when NACK is received in I ² C mode |
| Description | When the USCI_B module is configured as an I ² C master transmitter, the TXIFG is not reset after a NACK is received if the master is configured to send a restart (UCTXSTT = 1 and UCTXSTP = 0). |
| Workaround | Reset TXIFG in software within the NACKIFG interrupt service routine. |

USCI26
USCI Module
Function
 t_{buf} parameter violation in I²C multi-master mode

Description

In multi-master I²C systems, the timing parameter t_{buf} (bus free time between a stop condition and the following start) is not ensured to match the I²C specification of 4.7 μ s in standard mode and 1.3 μ s in fast mode. If the UCTXSTT bit is set during a running I²C transaction, the USCI module waits and issues the start condition on bus release, causing the violation to occur.

NOTE: It is recommended to check if UCBBUSY bit is cleared before setting UCTXSTT = 1.

Workaround

None

USCI28
USCI Module
Function

Timing of USCI interrupts may cause device reset due to automatic clear of an IFG.

Description

When certain USCI I²C interrupt flags (IFGs) are set and an automatic flag-clearing event on the I²C bus occurs, it results in an errant ISR call to the reset vector. This happens only when the IFG is cleared within a critical time window (~6 CPU clock cycles) after a USCI interrupt request occurs and before the interrupt servicing is initiated. The affected interrupts are UCBxTXIFG, UCSTPIFG, UCSTTIFG, and UCNACKIFG.

The automatic flag-clearing scenarios occur in the following situations:

- A pending UCBxTXIFG interrupt request is cleared on the falling SCL clock edge following a NACK.
- A pending UCSTPIFG, UCSTTIFG, or UCNACKIFG interrupt request is cleared by a following Start condition.

Workaround

- Poll the affected flags instead of enabling the interrupts.
or
- Ensure the above mentioned flag-clearing events occur after a time delay of 6 CPU clock cycles has elapsed since the interrupt request occurred and was accepted.
or
- At program start, check any applicable enabled IE bits such as UCBxTXIE, UCBxRXIE, UCSTTIE, UCSTPIE, or UCNACKIE for a reset (A PUC clears all of the IE bits of interest). If no PUC occurred, then the device ran into the above mentioned errant condition, and the program counter needs to be restored using an RETI instruction.

; ----- Workaround (3) example for TXIFG -----

NOTE: For assembly code, use code shown here and insert prior to user code.

```
main
    bit.b  #UCBxTXIE ,&IE2    ; if TXIE is set, errant call occurred
    jz     start_normal        ; if not start main program
    reti                                ; else return from interrupt call
start_normal
    ...                          ; Application code continues
```


NOTE: For C code, the workaround needs to be executed prior to the CSTARTUP routine. The steps for modifying the CSTARTUP routine are IDE dependent. Examples for Code Composer and IAR Embedded Workbench are shown.

IAR Embedded Workbench

1. The file cstartup.s43 is found at: ...\\IAR Systems\\<Current Embedded Workbench Version>\\430\\src\\lib\\430
2. Create a local copy of this file and link it to the project. Do not rename the file.
3. In the copy, insert the following code prior to stack pointer initialization:

```
#define IE2      (0x0001)
      BIT.B  #0x08,&IE2      ; if TXIE is set, errant call occurred
      JZ     Start_Normal    ; if not start main program
      RETI     ; else return from interrupt call
// Initialize SP to point to the top of the stack.
Start_Normal
      MOV     #SFE(CSTACK), SP
// Ensure that main is called.
```

Code Composer

1. The file boot.c is found at ...\\Texas Instruments\\<Current Code Composer Version>\\tools\\compiler\\MSP430\\lib\\rtssrc.zip
2. Extract the file from rtssrc.zip and create a local copy. Link the copy to the project. Do not rename this file.
3. In the copy, insert the following code prior to stack pointer initialization:

```
__asm("\t BIT.B\t  #0x08,&0x0001"); // if TXIE is set, errant call occurred
__asm("\t JZ\t      Start_Normal"); // if not start main program
__asm("\t RETI"); // else return from interrupt call
__asm("Start_Normal"); // insert label

/*----- */
/* Initialize stack pointer. Stack grows toward lower memory*/
/*----- */
```

USCI30
USCI Module
Function

I2C mode master receiver / slave receiver

Description

The USCI I2C module, when configured as a receiver (master or slave), performs a double-buffered receive operation. For example, in a transaction of two bytes, after the first byte is moved from the receive shift register to the receive buffer, the byte is acknowledged and the state machine allows the reception of the next byte.

If the receive buffer has not been cleared of its contents by reading the UCBxRXBUF register by the time the seventh bit of the following data byte is received, an error condition may occur on the I2C bus. Depending on the USCI configuration, the following may occur:

- If the USCI is configured as an I2C master receiver, an unintentional repeated start condition can be triggered or the master can switch into an idle state (I2C communication aborted). The reception of the current data byte is not successful in this case.
- If the USCI is configured as I2C slave receiver, the slave can switch to an idle state, stalling I2C communication. The reception of the current data byte is not successful in this case. The USCI I2C state machine notifies the master of the aborted reception with a NACK.

Note that the error condition described above occurs only within a limited window of the seventh bit of the current byte being received. If the receive buffer is read outside of this window (before or after), then the error condition does not occur.

Workaround

The error condition can be avoided by servicing the UCBxRXIFG in a timely manner. This can be done by (a) servicing the interrupt and ensuring UCBxRXBUF is read promptly or (b) using the DMA to automatically read bytes from receive buffer upon UCBxRXIFG being set.

OR

If the receive buffer cannot be read out in time, test the I2C clock line before the UCBxRXBUF is read out to ensure that the critical window has elapsed. This is done by checking if the clock line low status indicator bit UCSCLOW is set for at least three USCI bit clock cycles; that is, $3 \times t_{\text{BitClock}}$.

NOTE: The last byte of the transaction must be read directly from UCBxRXBUF. For all other bytes, follow the workaround.

Code flow for workaround:

1. Enter RX ISR for reading receiving bytes
2. Check if UCSCLOW.UCBxSTAT == 1
3. If no, repeat step 2 until set.
4. If yes, repeat step 2 for a time period $> 3 \times t_{\text{BitClock}}$, where $t_{\text{BitClock}} = 1/f_{\text{BitClock}}$
5. If window of $3 \times t_{\text{BitClock}}$ cycles has elapsed, it is safe to read UCBxRXBUF.

XOSC5
LFXT1 Module
Function

LF crystal failures may not be properly detected by the oscillator fault circuitry

Description

The oscillator fault error detection of the LFXT1 oscillator in low-frequency mode (XTS = 0) may not work reliably, causing a failing crystal to go undetected by the CPU; that is, OFIFG is not set.

Workaround

None

XOSC8
LFXT1 Module
Function

ACLK failure when crystal ESR is below 40 kΩ

Description

When ACLK is sourced by a low-frequency crystal with an ESR below 40 kΩ, the duty cycle of ACLK may fall below the specification; the OFIFG may become set or, in some instances, ACLK may stop completely.

Workaround

See the application report *XOSC8 Guidance* ([SLAA423](#)) for information regarding working with this erratum.

Appendix A Prior Versions

✓ The checkmark means that the issue is present in the specified revision.

| Devices | Rev: | BCL12 | BCL13 | CPU19 | FLASH19 | FLASH24 | FLASH27 | PORT12 | TA12 | TA16 | TA22 | USCI20 | USCI21 | USCI22 | USCI23 | USCI24 | USCI25 | USCI26 | USCI28 | USCI30 | XOSC5 | XOSC8 |
|-------------|------|-------|-------|-------|---------|---------|---------|--------|------|------|------|--------|--------|--------|--------|--------|--------|--------|--------|--------|-------|-------|
| MSP430F2112 | B | ✓ | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | A | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| MSP430F2122 | B | ✓ | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | A | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| MSP430F2132 | B | ✓ | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | A | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

A.1 Detailed Bug Description

BCL13

Basic Clock Module

Function

Exiting reset state with slow V_{CC} rise time

Description

When subject to very slow V_{CC} rise times, the device may enter a state in which the DCO does not oscillate. No JTAG access or program execution is possible, and the device remains in the reset state until the supply voltage is disconnected.

Workaround

Apply a V_{CC} power-on ramp ≥ 10 V/s under all power-on/power-cycle scenarios.

Revision History

| Changes from C Revision (January 2010) to D Revision | Page |
|--|--------------------|
| • Updated USCI21 description | 6 |
| • Added USCI30 | 10 |

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products

| | |
|-----------------------------|--|
| Audio | www.ti.com/audio |
| Amplifiers | amplifier.ti.com |
| Data Converters | dataconverter.ti.com |
| DLP® Products | www.dlp.com |
| DSP | dsp.ti.com |
| Clocks and Timers | www.ti.com/clocks |
| Interface | interface.ti.com |
| Logic | logic.ti.com |
| Power Mgmt | power.ti.com |
| Microcontrollers | microcontroller.ti.com |
| RFID | www.ti-rfid.com |
| RF/IF and ZigBee® Solutions | www.ti.com/lprf |

Applications

| | |
|-------------------------------|--|
| Communications and Telecom | www.ti.com/communications |
| Computers and Peripherals | www.ti.com/computers |
| Consumer Electronics | www.ti.com/consumer-apps |
| Energy and Lighting | www.ti.com/energy |
| Industrial | www.ti.com/industrial |
| Medical | www.ti.com/medical |
| Security | www.ti.com/security |
| Space, Avionics and Defense | www.ti.com/space-avionics-defense |
| Transportation and Automotive | www.ti.com/automotive |
| Video and Imaging | www.ti.com/video |
| Wireless | www.ti.com/wireless-apps |

TI E2E Community Home Page

e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2011, Texas Instruments Incorporated