

Implementing a Real-Time Clock on the MSP430

Mike Mitchell
Mixed Signal Products

ABSTRACT

This application report describes the implementation of a real-time clock (RTC) on the MSP430 using either Timer_A or the Watchdog Timer. The MSP430 family of microcontrollers from Texas Instruments, is a family of ultralow-power, 16-bit RISC microcontrollers with an advanced architecture and a rich peripheral set. The report gives a general overview of the MSP430 family, and discusses the real-time clock application in detail. It gives a detailed circuit diagram, two code examples, and a general discussion on accuracy and implementation issues.

Contents

1	Introduction	2
	1.1 Real-Time Clock Application	2
	1.2 MSP430 Clock Generation	2
2	Real-Time Clock Implementation	2
	2.1 Clock Generation Setup	3
	2.2 Timer Selection	3
	2.3 External Interface	3
	2.4 Circuit Description	3
	2.5 Current Consumption	4
	2.6 Accuracy Issues	4
	2.6.1 Crystal Accuracy and Selection	4
	2.7 Expandability	5
3	Discussion of the Code	5
4	Summary	5
5	References	5
Appendix A Code Examples		6
	A.1 Using the Watchdog Timer—RTC11xWD.s43 File	6
	A.2 Using Timer_A—RTC11xTA.s43 File	9

List of Figures

1	RTC Circuit Diagram	4
---	---------------------------	---

1 Introduction

The MSP430 family of microcontrollers from Texas Instruments (TI) is a family of ultralow-power, 16-bit RISC microcontrollers with an advanced architecture and a rich peripheral set. The architecture uses advanced timing and design features, as well as a highly orthogonal structure, to deliver a processor that is both powerful and flexible. The MSP430x1xx devices consume less than 350 μA in active mode when operating at 1 MHz in a typical 3-V system, and can wake up from a 2 μA standby mode to fully synchronized operation in less than 6 μs . These exceptionally low current requirements, combined with the fast wake-up time, enable users to build a system with minimum current consumption and maximum battery life.

Additionally, the MSP430 family has an abundant mix of peripherals and memory sizes enabling true system-on-a-chip designs. The peripherals include 12- and 14-bit A/D converters, slope A/D, multiple timers (some with capture/compare registers and PWM output capability), LCD driver, on-chip clock generation, H/W multiplier, USART, Watchdog Timer, GPIO, and others.

See <http://www.ti.com> for the latest device information and literature on the MSP430 family.

1.1 Real-Time Clock Application

Real time clocks (RTC) are used in a variety of applications—from watches and clocks to time-stamping events, to generating events. This report describes how to implement an RTC on an MSP430F1121 device. In some applications, an MSP430 implementing a real-time clock can be used to replace dedicated RTC devices, thus simplifying system design and reducing costs. This report shows examples on the use of Timer_A and the Watchdog Timer to implement the RTC; however, any other MSP430 timer can be used in a similar manner.

All MSP430 devices can implement an RTC. Additionally, system integration, power savings, and cost savings may be achieved by leveraging some of the peripherals from the MSP430 family. For example, a '33x device could function as an RTC, a system supervisor, an LCD driver, and a UART, integrating all four functions onto a single, ultralow-power device.

1.2 MSP430 Clock Generation

Internal clock generation is one of the most commonly misunderstood subjects by MSP430 users. The primary cause of confusion lies with how a stable system clock is produced. All MSP430 devices contain both a digitally-controlled RC-type oscillator and a crystal oscillator. The RC-type oscillator is typically used for the CPU clock and the crystal oscillator is typically used for the peripherals. In the real-time clock application, the crystal oscillator is used as the clock source for the timer/counter that serves as the time base (either Timer_A or the Watchdog Timer in this application report). Therefore, the instability issues that are common to RC-type oscillators are irrelevant.

2 Real-Time Clock Implementation

The general implementation of the RTC is simple. It consists of a timer/counter giving 1-second interrupts and a small CPU routine to count the interrupts. The CPU can sleep or perform other functions between interrupts. This application report and the code examples were developed using the MSP430F1121. The following sections describe the design.

2.1 Clock Generation Setup

The clock setup for the RTC implementation uses the LFXT1 oscillator in LF mode with a 32768-Hz crystal. The output of the LFXT1 oscillator is selected to source ACLK. ACLK is then selected as the clock source for the timer/counter that serves as the time base for the RTC.

The DCO generates the CPU clock, MCLK. The CPU actually runs asynchronously to the timer/counter peripheral. Accuracy of the RTC is not affected as long as the CPU is able to service each interrupt from the timer peripheral before the next interrupt arrives. See the MSP430x1xx Family user's guide [1] (available from <http://www.ti.com/sc/msp430>) and the MSP430x11x1 data sheet [2] for more detailed discussions on the Basic Clock Module and its setup requirements.

2.2 Timer Selection

The MSP430F1121 device contains two timers: Watchdog Timer and Timer_A. An example showing the use of each of these timers is included at the end of this report. In both examples the timer is configured to count continuously and give interrupts at exactly 1-second intervals. Since the timer is configured to use ACLK as its clock source, and ACLK runs at exactly the crystal frequency (32768 Hz), the timer simply counts up to 32767 and starts over at 0, giving an interrupt each time it reaches 32767. The CPU then simply counts the interrupts to form the RTC.

Any timer capable of generating a 1-second interrupt can be used with the clock routine in this application report. Furthermore, the clock routine can be adjusted to use any timer giving periodic interrupts. For example, if a timer gives an interrupt exactly every 0.5 seconds, the CPU can simply count two interrupts as one second, and so on.

2.3 External Interface

The code in this application report does not provide an external interface for the RTC. If the MSP430 implementation is being used to replace a dedicated RTC chip, there are several interfaces to choose from, including I²C, parallel, UART, serial, etc. TI has code modules to implement these interfaces on MSP430 devices already written and available for easy integration. Building a complete RTC based on the MSP430 is simply a matter of choosing an interface and integrating the applicable routine(s) with the RTC routine shown here. More detailed discussions of some of the possible interfaces can be found in the *MSP430 Family Application Report Book* [3] and in individual application reports.

2.4 Circuit Description

Figure 1 shows the circuit diagram of the RTC. Note that the only external component required is the 32768-Hz crystal.

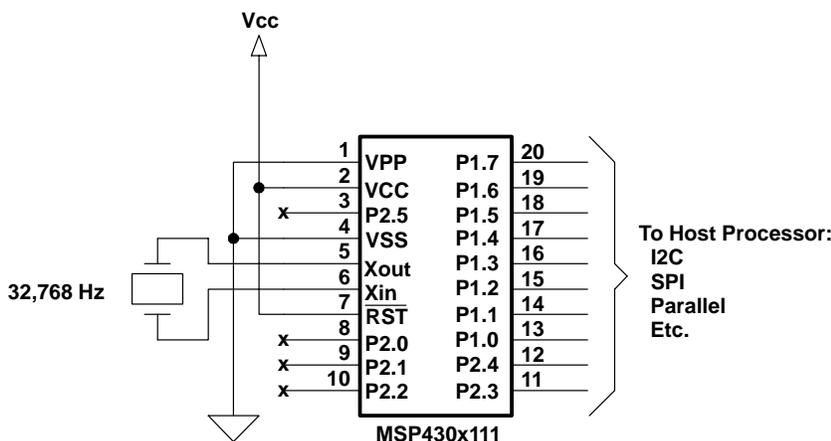


Figure 1. RTC Circuit Diagram

2.5 Current Consumption

The typical current consumption of the MSP430F11x1 in active mode is 300 μ A at 1 MHz and 3 V. Typical current consumption in low-power mode 3 (sleep mode) is 1.6 μ A at 3-V. The device wakes from low-power mode 3 in less than 6 μ s, and the clock routine executes in roughly 130 μ s. Given the ultralow current consumption and the very short time in active mode, the 'F11x1 operating as an RTC consumes very little current over time, and therefore maximizes battery life.

2.6 Accuracy Issues

The accuracy of the RTC depends on the accuracy of the crystal chosen for the crystal oscillator. This allows the user to purchase the appropriate crystal for the desired accuracy.

2.6.1 Crystal Accuracy and Selection

Crystal accuracy is affected primarily by two things: the crystal frequency tolerance, and the specified load capacitance.

The tolerance of the crystal is self-explanatory: the tighter the tolerance on the crystal frequency, the more accurate the RTC will be. For example, an RTC using a 30-ppm crystal is more accurate than one using a 200-ppm crystal.

The specified load capacitance of the crystal also affects the accuracy of the RTC. The load capacitance of a crystal is the amount of capacitance *required* by the crystal, not the amount *provided* by the crystal. The crystal requires the proper load capacitance to oscillate at its specified frequency. The 32768-Hz oscillator on all MSP430 devices has integrated load capacitors with a specified nominal capacitance of 12 pF. This provides an overall 6-pF load for the crystal because the capacitors add serially.

In addition, the stray capacitance of the board and pins add in a parallel fashion, thus increasing the capacitive load seen by the crystal. For example, 2 pF of stray capacitance is not uncommon on a circuit board. The addition of the 2 pF to the 6 pF yields a total capacitive load for the crystal of 8 pF. In some cases, depending on crystal specification and stray capacitances, parallel capacitors must be added to the circuit to provide the proper capacitive load for the crystal.

2.7 Expandability

MSP430x11x devices are ultralow-power, inexpensive microcontrollers that can be ideally suited to replace real-time clock devices. One of the key benefits of using an MSP430 as an RTC is the expandability of the MSP430 over a dedicated RTC device. All MSP430x11x devices contain a 16-bit RISC CPU, 16-bit Watchdog Timer, 16-bit Timer_A (with 3 capture/compare registers and analog comparator), a minimum of 128B RAM, a minimum of 2KB ROM, and a minimum of 14 GPIO pins. Not all of this functionality is available simultaneously, but it becomes clear that the device offers a great deal of flexibility over a dedicated RTC device.

In addition, the Timer_A module is capable of slope A/D conversion, PWM output, and UART operation up to 115,200 baud. The Watchdog Timer is capable of operating as a simple timer, and the GPIO pins and all peripherals have extensive interrupt capability.

3 Discussion of the Code

The code for the RTC application is straightforward; two examples are shown in Appendix A. In each example, there is an initialization routine, a main loop, a clock counting routine for hours, minutes, and seconds, and an interrupt service routine to handle the 1-second interrupts from the timer.

The initialization routine initializes various aspects of the MSP430. The timer module is setup to count continuously from 0 to 32,767 and to give an interrupt each time it reaches 32,767 and the Basic Clock Module is set up.

The mainloop is an endless loop that calls the clock counting routine every time the timer issues an interrupt, or puts the CPU in sleep mode otherwise. The timer continues counting while the CPU is asleep.

The timer interrupt service routine (ISR) manipulates the status register (SR) bits that were pushed onto the stack just prior to entering the ISR. This allows the CPU to be in active mode, rather than sleep mode, upon return from the ISR.

The clock counting routine counts each timer interrupt as 1 second. It counts seconds, minutes, and hours in binary-coded-decimal format. A more sophisticated counting routine that includes years, including leap years, can be found in the Basic Timer discussion of the *On-Chip Peripherals* chapter of the *MSP430 family Application Report Book* [3].

4 Summary

The MSP430 devices can implement an RTC or even replace a dedicated RTC chip, providing added flexibility and system integration.

5 References

1. *MSP430x1xx Family User' Guide*, literature number SLAU049
2. MSP430x11x1 data sheet, literature number SLAS241C
3. *MSP430 Family Application Report Book*, literature number SLAA024

Appendix A Code Examples

A.1 Using the Watchdog Timer—RTC11xWD.s43 File

```

;*****
; THIS PROGRAM IS PROVIDED "AS IS". TI MAKES NO WARRANTIES OR
; REPRESENTATIONS, EITHER EXPRESS, IMPLIED OR STATUTORY,
; INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
; FOR A PARTICULAR PURPOSE, LACK OF VIRUSES, ACCURACY OR
; COMPLETENESS OF RESPONSES, RESULTS AND LACK OF NEGLIGENCE.
; TI DISCLAIMS ANY WARRANTY OF TITLE, QUIET ENJOYMENT, QUIET
; POSSESSION, AND NON-INFRINGEMENT OF ANY THIRD PARTY
; INTELLECTUAL PROPERTY RIGHTS WITH REGARD TO THE PROGRAM OR
; YOUR USE OF THE PROGRAM.
;
; IN NO EVENT SHALL TI BE LIABLE FOR ANY SPECIAL, INCIDENTAL,
; CONSEQUENTIAL OR INDIRECT DAMAGES, HOWEVER CAUSED, ON ANY
; THEORY OF LIABILITY AND WHETHER OR NOT TI HAS BEEN ADVISED
; OF THE POSSIBILITY OF SUCH DAMAGES, ARISING IN ANY WAY OUT
; OF THIS AGREEMENT, THE PROGRAM, OR YOUR USE OF THE PROGRAM.
; EXCLUDED DAMAGES INCLUDE, BUT ARE NOT LIMITED TO, COST OF
; REMOVAL OR REINSTALLATION, COMPUTER TIME, LABOR COSTS, LOSS
; OF GOODWILL, LOSS OF PROFITS, LOSS OF SAVINGS, OR LOSS OF
; USE OR INTERRUPTION OF BUSINESS. IN NO EVENT WILL TI'S
; AGGREGATE LIABILITY UNDER THIS AGREEMENT OR ARISING OUT OF
; YOUR USE OF THE PROGRAM EXCEED FIVE HUNDRED DOLLARS
; (U.S.$500).
;
; Unless otherwise stated, the Program written and copyrighted
; by Texas Instruments is distributed as "freeware". You may,
; only under TI's copyright in the Program, use and modify the
; Program without any charge or restriction. You may
; distribute to third parties, provided that you transfer a
; copy of this license to the third party and the third party
; agrees to these terms by its first use of the Program. You
; must reproduce the copyright notice and any other legend of
; ownership on each copy or partial copy, of the Program.
;
; You acknowledge and agree that the Program contains
; copyrighted material, trade secrets and other TI proprietary
; information and is protected by copyright laws,
; international copyright treaties, and trade secret laws, as
; well as other intellectual property laws. To protect TI's
; rights in the Program, you agree not to decompile, reverse
; engineer, disassemble or otherwise translate any object code
; versions of the Program to a human-readable form. You agree
; that in no event will you alter, remove or destroy any
; copyright notice included in the Program. TI reserves all
; rights not specifically granted under this license. Except
; as specifically provided herein, nothing in this agreement
; shall be construed as conferring by implication, estoppel,
; or otherwise, upon you, any license or other right under any
; TI patents, copyrights or trade secrets.
;

```

```

; You may not use the Program in non-TI devices.
;*****
; RTC USING THE WATCHDOG TIMER
;
; Description: This program demonstrates a real time clock with hours
; minutes and seconds. The Mainloop immediately puts the MSP430 into LPM3.
; The Watchdog timer Interrupts at exactly 1-second intervals and returns the
; MSP430 to active mode which allows the program to complete
; the Mainloop. Mainloop calls a Clock routine then returns to LPM3.
;
; This program is written specifically for the MSP430F1121, but could
; easily be adapted for any other MSP430 device.
;
; Three registers are used to store the seconds, minutes and hour values.
;*****
#include      "msp430x11x1.h"          ; include std defs
;          RTC variables
#define      SEC      R13
#define      MIN      R14
#define      HR       R15
;-----
;          Program RESET
RSEG        CODE
;-----
RESET       MOV      #02FEh,SP        ; Initialize stackpointer
            CALL    #Setup           ; Prepare LCD and basic timmer
            ; Mainloop
Mainloop    BIS      #LPM3,SR        ; Set SR bits for LPM3
            CALL    #Clock           ; Update Clock
            JMP     Mainloop         ; Endless Loop
;-----
; Clock: Update clock SEC and MIN and HR
;
; Originally written by Lutz Bierl.
;
; This is the routine that counts the hours, minutes and seconds.
; It can be used with any counter peripheral on any MSP430 device,
; as long as it is executed once per second.
;
; This particular routine is very basic. It only counts minutes, seconds
; and hours as BCD numbers. Hex values could also be used if desired.
;
; Refer to the MSP430 Application Report Book for more extensive clock
; routines that include counters for years, adjustments for leap years, etc.
;-----
Clock       SETC      ; Set Carry bit.
            DADC.b  SEC      ; Increment seconds decimally
            CMP.b   #060h,SEC ; One minute elapsed?
            JLO    Clockend  ; No, return
            CLR.b   SEC      ; Yes, clear seconds
            DADC.b  MIN      ; Increment minutes decimally
    
```


A.2 Using Timer_A—RTC11xTA.s43 File

```

; THIS PROGRAM IS PROVIDED "AS IS". TI MAKES NO WARRANTIES OR
; REPRESENTATIONS, EITHER EXPRESS, IMPLIED OR STATUTORY,
; INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS
; FOR A PARTICULAR PURPOSE, LACK OF VIRUSES, ACCURACY OR
; COMPLETENESS OF RESPONSES, RESULTS AND LACK OF NEGLIGENCE.
; TI DISCLAIMS ANY WARRANTY OF TITLE, QUIET ENJOYMENT, QUIET
; POSSESSION, AND NON-INFRINGEMENT OF ANY THIRD PARTY
; INTELLECTUAL PROPERTY RIGHTS WITH REGARD TO THE PROGRAM OR
; YOUR USE OF THE PROGRAM.
;
;
; IN NO EVENT SHALL TI BE LIABLE FOR ANY SPECIAL, INCIDENTAL,
; CONSEQUENTIAL OR INDIRECT DAMAGES, HOWEVER CAUSED, ON ANY
; THEORY OF LIABILITY AND WHETHER OR NOT TI HAS BEEN ADVISED
; OF THE POSSIBILITY OF SUCH DAMAGES, ARISING IN ANY WAY OUT
; OF THIS AGREEMENT, THE PROGRAM, OR YOUR USE OF THE PROGRAM.
; EXCLUDED DAMAGES INCLUDE, BUT ARE NOT LIMITED TO, COST OF
; REMOVAL OR REINSTALLATION, COMPUTER TIME, LABOR COSTS, LOSS
; OF GOODWILL, LOSS OF PROFITS, LOSS OF SAVINGS, OR LOSS OF
; USE OR INTERRUPTION OF BUSINESS. IN NO EVENT WILL TI'S
; AGGREGATE LIABILITY UNDER THIS AGREEMENT OR ARISING OUT OF
; YOUR USE OF THE PROGRAM EXCEED FIVE HUNDRED DOLLARS
; (U.S.$500).
;
; Unless otherwise stated, the Program written and copyrighted
; by Texas Instruments is distributed as "freeware". You may,
; only under TI's copyright in the Program, use and modify the
; Program without any charge or restriction. You may
; distribute to third parties, provided that you transfer a
; copy of this license to the third party and the third party
; agrees to these terms by its first use of the Program. You
; must reproduce the copyright notice and any other legend of
; ownership on each copy or partial copy, of the Program.
;
; You acknowledge and agree that the Program contains
; copyrighted material, trade secrets and other TI proprietary
; information and is protected by copyright laws,
; international copyright treaties, and trade secret laws, as
; well as other intellectual property laws. To protect TI's
; rights in the Program, you agree not to decompile, reverse
; engineer, disassemble or otherwise translate any object code
; versions of the Program to a human-readable form. You agree
; that in no event will you alter, remove or destroy any
; copyright notice included in the Program. TI reserves all
; rights not specifically granted under this license. Except
; as specifically provided herein, nothing in this agreement
; shall be construed as conferring by implication, estoppel,
; or otherwise, upon you, any license or other right under any
; TI patents, copyrights or trade secrets.
;
; You may not use the Program in non-TI devices.
    
```

```

;*****
;
; RTC USING TIMER_A
;
; Description: This program demonstrates a real time clock with hours
; minutes and seconds. The Mainloop immediately puts the MSP430 into LPM3.
; The Timer_A Interrupts at exactly 1-second intervals and returns the
; MSP430 to active mode which allows the program to complete
; the Mainloop. Mainloop calls a Clock routine then returns to LPM3.
;
; This program is written specifically for the MSP430F1121, but could
; easily be adapted for any other MSP430 device.
;
; Three registers are used to store the seconds, minutes and hour values.
;*****
;
#include      "msp430x11x1.h"          ; include std defs
;          RTC variables
#define      SEC      R13
#define      MIN      R14
#define      HR       R15
;-----
;          Program RESET
;          RSEG      CODE
;-----
RESET      MOV      #02FEh,SP          ; Initialize stackpointer
           CALL     #Setup             ; Prepare LCD and basic timer
           ; Mainloop
Mainloop   BIS      #LPM3,SR          ; Set SR bits for LPM3
           CALL     #Clock             ; Update Clock
           JMP      Mainloop          ; Endless Loop
;-----
; Clock: Update clock SEC and MIN and HR
;
; Originally written by Lutz Bierl.
;
; This is the routine that counts the hours, minutes and seconds.
; It can be used with any counter peripheral on any MSP430 device,
; as long as it is executed once per second.
;
; This particular routine is very basic. It only counts minutes, seconds
; and hours as BCD numbers. Hex values could also be used if desired.
;
; Refer to the MSP430 Application Report Book for more extensive clock
; routines that include counters for years, adjustments for leap years, etc.
;-----
Clock      SETC      ; Set Carry bit.
           DADC.b   SEC          ; Increment seconds decimally
           CMP.b    #060h,SEC    ; One minute elapsed?
           JLO     Clockend      ; No, return
           CLR.b    SEC          ; Yes, clear seconds

```

```

        DADC.b  MIN                ; Increment minutes decimally
        CMP.b   #060h,MIN          ; Sixty minutes elapsed?
        JLO    Clockend           ; No, return
        CLR.b   MIN                ; yes, clear minutes
        DADC.b  HR                 ; Increment Hours decimally
        CMP.b   #024h,HR          ; 24 hours elapsed?
        JLO    Clockend           ; No, return
        CLR.b   HR                 ; yes, clear hours
Clockend  RET                     ;
;-----
;      Setup: Configure Modules and Control Registers
;-----

Setup     MOV     #WDTPW+WDTHOLD,&WDTCTL ; Stop Watchdog Timer

setupTA   MOV     #TASSEL0+TACLRL,&TACTL ; ACLK for Timer_A.
          BIS     #CCIE,&CCTL0         ; Enable CCR0 interrupt.
          MOV     #07FFFh,&CCR0        ; load CCR0 with 32,767.
          BIS     #MC0, &TACTL        ; start TA in "up to CCR0" mode

ClearRTC  MOV.b   #00h,SEC          ; Clear SEC
          MOV.b   #00h,MIN          ; Clear MIN
          MOV.b   #00h,HR           ; Clear HR
          EINT                    ; Enable interrupts
          RET                     ; Done with setup.
;-----
;  Timer_A ISR:
;  CPU is simply returned to active state on RETI by manipulated the SR
;  bits in the SR value that was pushed onto the stack.
;  CCR0 IFG is reset automatically
;-----
CCR0INT   BIC     #LPM3,0(SP)        ; Clear SR LPM3 Bits, on top of stack
          RETI                    ;
;-----
          RSEG    INTVEC             ; Interrupt vectors
;-----
          DW     RESET              ;
          DW     CCR0INT            ; Timer_A (CCIFG0)
          DW     RESET              ; Watchdog Timer
          DW     RESET              ;
          DW     RESET              ;
          DW     RESET              ;
          DW     RESET              ; NMI, Osc. fault
          DW     RESET              ; POR, ext. Reset, Watchdog
          END
    
```

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 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.

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.

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265