

AN-2260 MSP430 Interface to LMP91050 Code Library

ABSTRACT

The MSP430™ is an ideal microcontroller solution for low-cost, low-power precision sensor applications because it consumes very little power. The LMP91050 is a programmable integrated analog front end (AFE) for thermopile sensors, as typically used in nondispersive infrared (NDIR) applications. This library provides functions to facilitate the interfacing of any MSP430 device to a LMP91050. Any device within the MSP430 family can be used with this library, made possible by hardware abstraction. Similarly, any SPI-capable interface module within the MSP430 family is supported by the library. This allows the designer maximum flexibility in choosing the best MSP430 device for the application. This document provides descriptive information and instructions for using the library either for demonstration purposes or implementation into a project. This is the recommended starting point for developing software for the LMP91050 and MSP430 combination. The software examples have been developed for the MSP430F5528 breakout board, but can easily be ported to another hardware platform.

Source code discussed in this application report can be downloaded from the LMP91050 product folder.

Contents

1	Introduction	2
2	Purpose and Scope	2
3	File Organization	2
4	Functions	4
5	Using the Software	5
	5.1 Prerequisites	5
	5.2 Getting Started	5
	5.3 Adapting the Demo Project to Other Hardware	8
	5.4 Using the Library with an Application	9
6	References	9

List of Figures

1	Code Library Stack	4
2	LMP91050SDEVAL to MSP430 Connection Diagram	6
3	LMP91050SDEVAL Jumper Settings	7
4	MSP-TS430RGC64USB Jumper Settings	8

List of Tables

1	Hardware Definition Files	2
2	Library Code	3
3	Demo Applications Included With the Library	3
4	Register Access and Control Functions Provided by the Library	4
5	LMP91050SDEVAL Jumper Settings	7

1 Introduction

This application report describes different ways to interface and use the TI LMP91050 devices with an MSP430. The accompanying software contains a function library allowing quick prototyping of NDIR sensor AFE setup and control. The software provided in this library is a starting point for developers wanting to get the most out of the MSP430 and the LMP91050 sensor AFE devices.

The LMP91050 is a configurable sensor AFE for thermopile sensors. It provides a complete signal path solution between a sensor and microcontroller that generates an output voltage proportional to the thermopile voltage. The LMP91050 is equipped with a slave serial peripheral interface (SPI) port, through which it can communicate with an MSP430 and is an optimal match for the MSP430 ultra low power microcontrollers. The MSP430 is a great fit for applications where power conservation is a priority. The many power-saving mechanisms designed into the MSP430 make it ideal for such applications.

2 Purpose and Scope

To aid in interfacing these devices, TI has produced a code library that significantly reduces the need to write low-level interface functions. It provides a boost in the development of an MSP430/LMP91050-based product, saving time and allowing quick progression to the application-specific aspects of the project. This library is designed to be used with any MSP430 device. Since a SPI master can be implemented using one of many peripherals within the MSP430 family, and since the peripherals available may differ by device and application, library calls are provided for each of these interfaces. The chosen interface is selected by assigning a value to a system variable, which causes the compiler to conditionally include the appropriate function calls. As such, application code utilizing the library remains portable between various MSP430 devices, with minimal modification required.

Several complete example application projects are provided with the library. The purpose of these projects is to demonstrate use of the library. It is not intended as a comprehensive guide to using the LMP91050, and it does not make use of all the features of these devices. It does, however, use all the register access functions provided by the library.

3 File Organization

The library has been implemented with modular hardware abstraction. There is a header file specific to each of the hardware components (LMP91050, MSP430, and the board). The hardware definition header files are shown in [Table 1](#). [Table 2](#) shows the library code files and its header, and [Table 3](#) shows the demonstration applications that accompanies the library.

Table 1. Hardware Definition Files

Filename	Description
<i>TI_LMP91050.h</i>	Definitions specific to the LMP91050 device, including register bit definitions and commonly-used masks for use with these registers.
<i>TI_MSP430.h</i>	Definitions specific to the MSP430 device; primarily, the pins used in the SPI interface. Definitions for USART0/1, USCI_A0/1/2/3, USCI_B0/1/2/3, USI and bit banging are included. Also, labels are defined for use with the system variable <i>TI_LMP91050_SER_INTF</i> . This label selects the modules to be used for accessing the LMP91050 SPI interface.
<i>TI_MSP430_hardware_board.h</i>	Definitions specific to the board being used; that is, the connections between the MSP430 and LMP91050SDEVAL, such as the SPI connections to the onboard ADC141S628, MOSI_EN, chip select, and LED pins. SPI connections to the LMP91050 are not defined here because they are defined inherently within <i>TI_MSP430.h</i> . The system variable <i>TI_LMP91050_SER_INTF</i> is defined in this file.

Table 2. Library Code

Filename	Description
<i>TI_MSP430_spi_USCIA0_5xx.c</i>	Function for accessing the LMP91050 registers via SPI_USCIA0 module from the MSP430 5xx family
<i>TI_MSP430_other_spi_modules\</i>	Functions for accessing the LMP91050 registers via other MSP430 SPI modules like SPI_USART0, SPI_USART1, SPI_USCIA1, etc.
<i>TI_MSP430_spi_USCIA1_5xx.c</i>	
<i>TI_MSP430_spi_USCIA2_5xx.c</i>	
<i>TI_MSP430_spi_USCIA3_5xx.c</i>	
<i>TI_MSP430_spi_USCIB0_5xx.c</i>	
<i>TI_MSP430_spi_USCIB1_5xx.c</i>	
<i>TI_MSP430_spi_USCIB2_5xx.c</i>	
<i>TI_MSP430_spi_USCIB3_5xx.c</i>	
<i>TI_MSP430_spi_USCIA0.c</i>	
<i>TI_MSP430_spi_USCIA1.c</i>	
<i>TI_MSP430_spi_USCIB0.c</i>	
<i>TI_MSP430_spi_USCIB1.c</i>	
<i>TI_MSP430_spi_USART0.c</i>	
<i>TI_MSP430_spi_USART1.c</i>	
<i>TI_MSP430_spi_USI.c</i>	
<i>TI_MSP430_spi_BITBANG.c</i>	
<i>TI_MSP430_spi_eUSCIA0_FR57xx.c</i>	
<i>TI_MSP430_spi_eUSCIA1_FR57xx.c</i>	
<i>TI_MSP430_spi_eUSCIB0_FR57xx.c</i>	
<i>TI_MSP430_spi.h</i>	Function declarations for <i>TI_MSP430_spi_*.c</i>

Table 3. Demo Applications Included With the Library

Demo Application	Filename	Description
Application1: Read/Write LMP91050 Registers	<i>demo-app01\main.c</i>	Application code with functions to demonstrate read/write of the LMP91050 register
	<i>demo-app01\TI_LMP91050 _register_settings.h</i>	Application specific initialization values for the LMP91050 registers
Application2: Common mode voltage measurement using the onboard ADC141S628	<i>demo-app02\main.c</i>	Application code with functions to measure the default common mode voltage. Adjust the internal DAC code and measure output level shift.
	<i>demo-app02\adc14s628.c</i>	Application code with functions to initialize and read the ADC141S628 data output
	<i>demo-app02\TI_LMP91050 _register_settings.h</i>	Application specific initialization values for the LMP91050 registers
Application3: Common mode voltage measurement using the ADC12 module in MSP430F5528	<i>demo-app03\main.c</i>	Application code with functions to measure the default common mode voltage. Adjust the internal DAC code and measure output level shift.
	<i>demo-app03\TI_LMP91050 _register_settings.h</i>	Application specific initialization values for the LMP91050 registers

NOTE: The register settings values for *TI_LMP91050_register_settings.h* can easily be obtained from the "Register configuration file" saved from Sensor AFE Software [2]. The demo application code reads the values stored in the register settings file to initialize the LMP91050 device register.

Figure 1 shows a stack diagram of the library. Note that one of the files displayed in the stack is the standard definition file for the specific MSP430 device being used. This file is included with the development environment being used to create the MSP430 software.

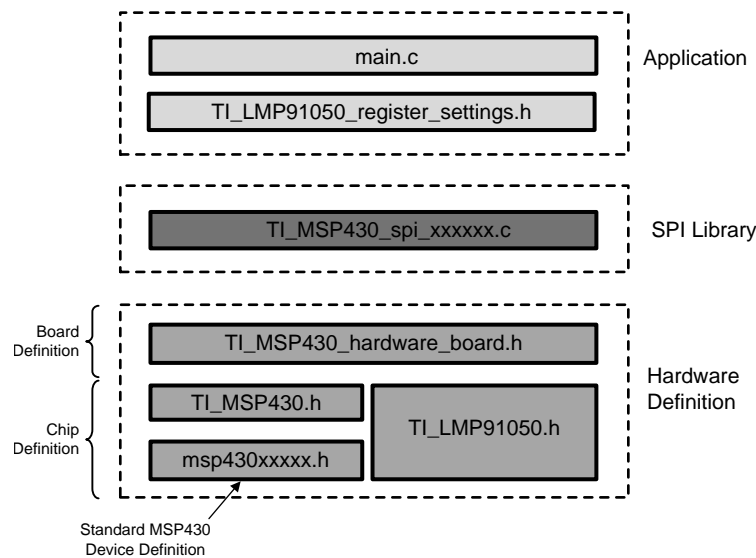


Figure 1. Code Library Stack

4 Functions

Table 4 shows the SPI register-access functions provided in the library, with a brief description.

Table 4. Register Access and Control Functions Provided by the Library

Function Name	Description
<code>void TI_LMP91050_SPISetup (void)</code>	Configures the SPI port assigned by the <code>TI_LMP91050_SER_INTF</code> system variable. Must be called before calling any of the other functions.
<code>void TI_LMP91050_SPIWriteReg (uint8_t addr, uint8_t value)</code>	Writes "value" to the LMP91050 configuration register at address "addr".
<code>uint8_t TI_LMP91050_SPIReadReg (uint8_t addr)</code>	Reads a single register at address "addr" and returns the 8-bit value read.

A version of these functions is provided for all MSP430 peripherals that are capable of communicating using the SPI protocol.

These peripherals are:

- USART0 for 1xx, 2xx, and 4xx families
- USART1 for 1xx, 2xx, and 4xx families
- USCI_A0 for 5xx and 6xx families
- USCI_A1 for 5xx and 6xx families
- USCI_A2 for 5xx and 6xx families
- USCI_A3 for 5xx and 6xx families
- USCI_B0 for 5xx and 6xx families
- USCI_B1 for 5xx and 6xx families
- USCI_B2 for 5xx and 6xx families
- USCI_B3 for 5xx and 6xx families
- USCI_A0 for 2xx and 4xx families
- USCI_A1 for 2xx and 4xx families

- USCI_B0 for 2xx and 4xx families
- USCI_B1 for 2xx and 4xx families
- USI for G2xx value series family
- Bitbang using GPIO pins
- eUSCI_A0 for FRAM 57xx family
- eUSCI_A1 for FRAM 57xx family
- eUSCI_B0 for FRAM 57xx family

5 Using the Software

5.1 Prerequisites

To successfully compile, download and run the software described in this document, the following materials are needed:

- MSP430 Target Board MSP-TS430RGC64USB Board
- LMP91050 Evaluation Board LMP91050DEVAL
- MSP430 USB Debugging Interface MSP430-FET430UIF
- IAR Embedded Workbench or TI Code Composer Studio™ for MSP430

The software can be adapted to run on other MSP430 hardware boards as well. For instructions, see [Section 5.3](#).

A free, code size limited, but fully functional edition of IAR Embedded Workbench (IAR Kickstart) is available from the IAR Systems website (www.iar.com) or from the TI MSP430 software tools page:

http://www.ti.com/lstds/ti/microcontroller/16-bit_msp430/msp430_software_landing_page.

As an alternative to IAR Embedded Workbench, it would be possible to use Code Composer Studio. A trial edition of the Code Composer Studio is available from the TI MSP430 software tools page.

5.2 Getting Started

Follow these simple steps to get your application up and running:

1. Install IAR Workbench.
2. Download the source code for this application note and unzip the files to your working directory.
3. Open IAR Embedded Workbench and create a new project:
 - (a) Select Project → Create new project
 - (b) Select tool chain MSP430
 - (c) Base the project on the empty project template
 - (d) Save (you will also be asked to save the current workspace)
4. Add the following C files to the project from the software that you downloaded and unzipped in step 2:
 - (a) All C files from the *code\library* folder
 - (b) All C files from the *code\library\TI_MSP430_other_spi_modules* folder
 - (c) All C files from the *code\demo-application-examples\demo-app01* folder
5. Open the "options" dialog for the new project by right clicking the project name in the workspace window. A window should appear:
 - (a) Under "General options", select the MSP device. For the MSP-TS430RGC64USB target board, use MSP430F5528.
 - (b) Under "C/C++ compiler", click on the preprocessor pane.
 - (i) The "Ignore standard include directories" tick box should not be ticked.
 - (ii) In the "Additional include directories", add include paths telling the compiler where to find the header files included by the C files. You should add the *\$PROJ_DIR\code\include* folder.
 - (c) Select "FET Debugger" from the "Driver" drop down list under "Debugger".

- (d) Choose the connection type of the FET tool (Texas Instruments USB-IF). Leave the rest of the settings as is, under "FET Debugger", in the "Connection" section.
6. Click "OK" to close the options window.
7. Select "Project → Rebuild All". There should be no errors or warnings when IAR rebuilds the executables (if not done already, you will also be asked to save the current workspace).
8. The configuration of the hardware definition files in the library as distributed by TI is for an MSP430F5528 equipped board. The system variable `TI_LMP91050_SER_INTF` defined within `TI_hardware_board.h` identifies `USCIA0_5xx` as the connected SPI port to control the LMP91050 . Peripheral pinouts can change slightly between individual MSP430 devices and families. For this reason `TI_MSP430.h` identifies the pins that correspond to a peripheral for any given device.
9. Connect the LMP91050 Evaluation Board SPI interface lines to the MSP430 target board SPI port as shown in [Figure 2](#). Note that the USCIA0 pins of the MSP430 are used to control the LMP91050 SPI pins.
10. Also, connect the ADC141S628 SPI interface lines of the LMP91050SDEVAL to the MSP430 target board SPI port as shown in [Figure 2](#). The USCIB1 pins of the MSP430 are used to control the ADC141S628 SPI pins.

NOTE: An external +5V supply is required for generating the ADC141S628 voltage reference of +4.096 V. If on-chip ADC12 module of the MSP430F5528 is used (instead of the ADC141S628) as in demo-app03 example, the external +5 V supply is not required.

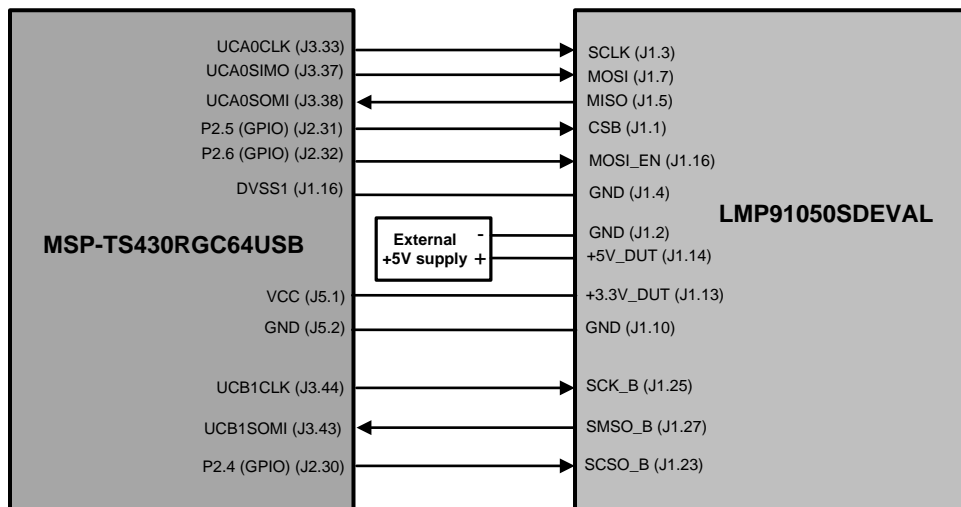


Figure 2. LMP91050SDEVAL to MSP430 Connection Diagram

11. The LMP91050 Evaluation Board jumper connections can be seen in [Figure 3](#) and [Table 5](#). Jumpers not shown can be left unpopulated.

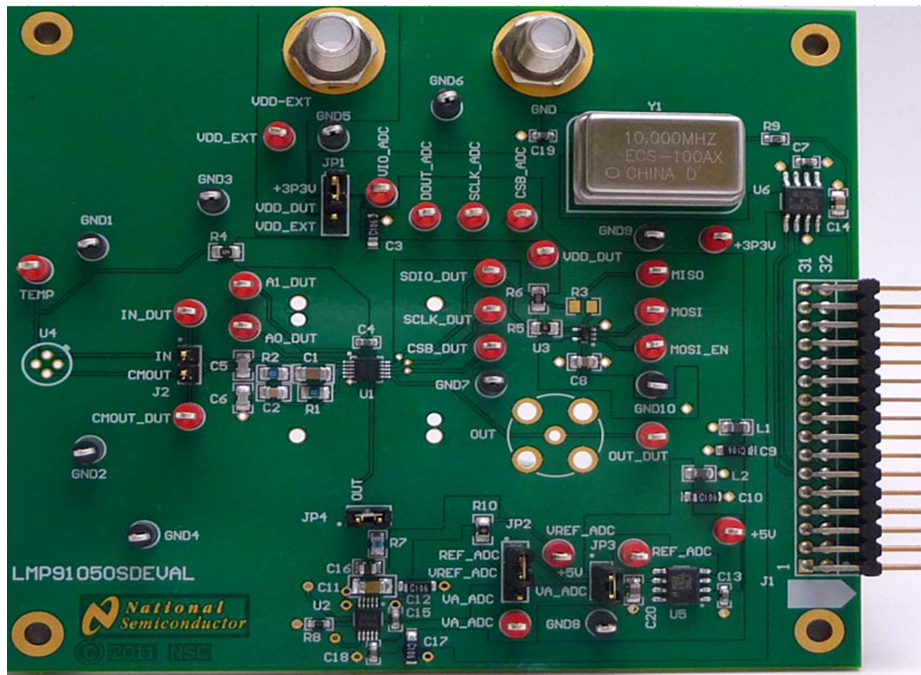


Figure 3. LMP91050SDEVAL Jumper Settings

Table 5. LMP91050SDEVAL Jumper Settings

Jumpers	Configuration	Purpose
JP1	1-2 shorted	Supply voltage (3.3 V) of the LMP91050 provided from the SPIO4 connector
JP2	1-2 shorted	Connect the onboard voltage reference (4.096 V) to the VREF pin of the LMP91050
JP3	1-2 shorted	+5 V analog supply for the onboard voltage reference
JP4	1-2 shorted	Connect the VOUT pin of the LMP91050 to the onboard ADC141S628 input
J2	1-2 shorted	Connect the LMP91050 IN to CMOUT for easy evaluation

12. Attach the MSP430 FET to your PC. If you are running Windows and using the USB FET tool for the first time, you will be asked to install some drivers for the tool. For Windows, they are located in `$IAR_INSTALL_DIR$430\drivers\TIUSBFET`.
13. Attach the MSP430 FET to the MSP430 target board using the JTAG connector. The V_{CC} power select jumper JP3 should be set to 1-2 (int) where the board is powered from the FET alone.

14. The MSP430 target board jumper connections can be seen in [Figure 4](#).

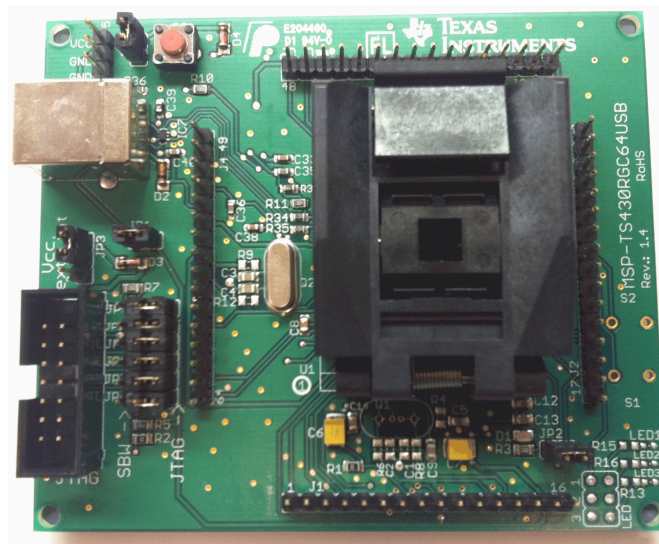


Figure 4. MSP-TS430RGC64USB Jumper Settings

15. Select Project → Debug. IAR will now establish a connection with the target MCU, download the application, and program the MSP430. The debugger will be started, halting the target at *main* ().
16. Demo_app01 is a simple example that demonstrates the SPI calls to write and successfully read back the LMP91050 register. The onboard LED on the MSP430 target board is setup to blink continuously if the value read back matches the value written.
17. Steps 3 to 15 can be followed to exercise other demo applications included with the library as well.

5.3 Adapting the Demo Project to Other Hardware

The procedure for adapting this code to other hardware is as follows:

1. Edit the pin assignments within *TI_MSP430.h* for the interface modules being used. It is not necessary to modify the pins for the interfaces not selected for use with the SPI bus, as they will not be referenced by the library. The labels being referenced in the `#define` assignments will be drawn from the standard definition file (*mcp430.h*) listed at the top of *TI_MSP430.h*.
2. Edit the pin assignments in *TI_MSP430_hardware_board.h*, taking into account all the necessary connections on the board being used. The assigned labels are drawn from the standard definition file (*mcp430.h*) listed at the top of *TI_MSP430.h*.

NOTE: The onboard ADC141S628 SPI interface lines are defined here.

3. Assign the proper values to *TI_LMP91050_SER_INTF* in *TI_MSP430_hardware_board.h*. The labels available for assignment can be found at the bottom of *TI_MSP430.h*.
4. Appropriately set up the function to configure the system clock source and clock rate. This depends on your hardware and the particular MSP430 MCU in use.
5. The default setup is to use the onboard ADC141S628 for converting the LMP91050 output. If the ADC141S626 is not used, appropriately set up the functions to configure and convert the LMP91050 output. This depends on your hardware and the particular MSP430 MCU in use.
6. Make sure the physical hardware connections between the MSP430 target board and the LMP91050SDEVAL are modified according to the pin assignments above.

After making these changes, rebuild the project and download the code image. The application should function as described earlier.

5.4 Using the Library with an Application

The same procedure as described in [Section 5.3](#) should be applied in order to adapt the library to the new hardware.

The function `TI_LMP91050_SPISetup ()` should always be called after a POR event within the MSP430. After this the access of registers is straightforward.

6 References

1. *LMP91050 Configurable AFE for Nondispersive Infrared (NDIR) Sensing Applications Data Sheet* ([SNAS517](#))
2. *LMP91050 Sensor AFE Software Download* ([SNAC004](#))
3. *User's Guide for the LMP91050 Evaluation Board With Sensor AFE Software* ([SNAU119](#))
4. *MSP430F551x/MSP430F552x Mixed Signal Microcontroller Data Sheet* ([SLAS590](#))
5. *MSP430x5xx/MSP430x6xx Family User's Guide* ([SLAU208](#))
6. MSP430F55xx 64-Pin Target Board MSP-TS430RGC64USB
7. MSP430 USB Debugging Interface MSP-FET430UIF

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as "components") are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

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

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

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license 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 significant portions 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. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

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

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI's goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or "enhanced plastic" are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have **not** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

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
OMAP Applications Processors	www.ti.com/omap
Wireless Connectivity	www.ti.com/wirelessconnectivity

Applications

Automotive and Transportation	www.ti.com/automotive
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
Video and Imaging	www.ti.com/video

TI E2E Community

e2e.ti.com