User's Guide for Sonic MDIO Software

This guide explains the usage of the TI Sonic MDIO Software to provide a means of communication with an MDIO compatible device. It communicates through the serial port of the host PC. The signals get translated from a serial port format to the MDIO-specified format by a required MDIO interface board, also developed by Texas Instruments. The program provides read and write access to the MDIO port on the device and basic scripting functions to allow the user to write scripts that contain a sequence of MDIO commands for more complex tasks.

Contents

List of Figures

1 Introduction

The Sonic MDIO Software was written to give the user the ability to communicate with a MDIO-compatible device. The user tells the software how to communicate with the device by writing scripts of commands. These scripts are called sequences. The program allows the user to write a main script, and other sub-scripts, called macros. This guide explains the graphical user interface (GUI) and the sequence language.

2 Sequence Language

This section is a language reference for the MDIO software. First, it will discuss the conventional use of the language. This is followed by a definition of all the valid sequence commands.

2.1 Coding Guide

The MDIO software allows the user to write scripts that perform the MDIO commands. The scripts are simply a list of commands that are executed sequentially. These scripts are called sequences or macros.

The program uses line-by-line execution. The main sequence can operate independently, or call other macros. The macros can be considered subroutines because they are a sequence of commands that may be executed multiple times.

The following example of code shows a main sequence and two defined macros.

```
EXAMPLE.TXT

START
//THIS IS THE MAIN SEQUENCE
READ(8000)
WRITE(8002,ABCD)
READ(8002)
THIRD MACRO
DISPLAY(EXECUTING MAC2 NOW...)
MAC2
DISPLAY(READ 8000 ONCE MORE)
READ(8000)
STOP

MACDEF MACRO 1
//THIS IS THE DEFINITION OF "MACRO 1"
START
READ(8230,0000)
DISPLAY(REGISTER 8230 WAS JUST READ)
WRITE(8000,1234)
STOP

MACDEF MAC2
//THIS IS THE DEFINITION OF "MAC2"
START
DISPLAY(MAC2 NOW EXECUTING...)
MACRO 1
STOP

MACDEF THIRD MACRO
//THIS IS THE DEFINITION OF "THIRD MACRO"
START
DISPLAY(THIRD MACRO NOW EXECUTING...)
STOP
```

At this point, do not be concerned with what the commands are doing or their syntax. This is explained later in the chapter. For now, concentrate on the modularity of the code. We see that all sequences (main or macros) contain a START and STOP command. Also, each macro is identified (or defined) by a “macdef <macroname>" statement preceding the macro.

Any sequence can call (or execute) another sequence by listing the macro’s name in the sequence. An example is the main sequence calls “MACRO 1” on the fifth line. The only exception is that the main sequence cannot be called by a macro.

When the sequence is executed within the software, it executes the main sequence only. If a macro is not called, then it is not executed. Defined macros do not have to be executed. In the example, the last macro “THIRD MACRO” is defined, but never called. So for this example, “running the sequence” means that the software would execute the main sequence. It contains four commands followed by a call for “MAC2” (the second defined macro). At this point, it would pause execution of the main sequence, and jump to the “MAC2” macro. "MAC2" has one command, and then calls “MACRO 1”, so “MACRO 1” is executed. After “MACRO 1” is completed, execution returns to “MAC2”, which ends at this point with the STOP command. Execution is returned back to the main sequence. It completes the main sequence when it reaches the STOP command, and execution is then completed.

NOTE: Macro names cannot include a sequence command anywhere within the name. For example, a macro name of SCRATCHPADREAD is not correct and would cause an error message, if used.

2.2 Sequence Command Reference

This section lists and defines all of the supported commands within a sequence. In this section, arguments are surrounded with “<" and ">" (for example: <mask>). If an argument is optional, it is surrounded with “[“ and “]” (for example: [<mask>]).

// This is the formal indication of a comment used to comment code. Text following a “//” until end of line is not printed to the output.

Syntax: //<text>
Example: //This is a comment in the code

NOTE: Text contained inside a sequence will be ignored if it is not a valid command, macro name, or part of a valid command. Care should be taken to use the “//” command to mark all comments. Anything other than valid commands or comments preceded by the “//” command should be removed to prevent the risk of the software from interpreting the text as a possible command.

START – This command defines the start point of execution. Any text before a START command is ignored.

Syntax: START

STOP – This command defines the end point of execution. It also defines the end point of a loop (see REPEAT command definition). If a sequence contains a REPEAT command, the first STOP defines the end of the loop, and another STOP command is needed to end the sequence. See the following example:

```
Syntax:   STOP
Example:  START
              DISPLAY(now in sequence)
              REPEAT 10
                    DISPLAY(now in loop)
                    STOP //Ends the loop
              DISPLAY (now back in sequence)
          STOP //Ends the sequence
```

DISPLAY – This command prints text to the output window. The output text color is blue.

Syntax: DISPLAY(<text>)
Example: DISPLAY(Hello World!) *Prints "Hello World!" in blue text in output window.*

READ – This command executes an MDIO read on the given register address. This command has two optional arguments that change the way the output displays. If the optional arguments are not given, then the READ command simply reports the resulting data from the given register address (example 1). If a valid hexadecimal value is given for the <expected data> argument, then the READ command assumes a <mask> value of "FFFF" and "read verify" of all 16 bits. The READ command reports the resulting data and expected data in green or red text. Green text means they compared equally, and red text means the resulting hexadecimal value was different than the expected data. If all three arguments are given, the READ performs a "read verify", but only on the bits specified in the <mask> argument.

Syntax: READ(<regaddr>, [<expected data>], [<mask>])
Example: READ(8000)
READ(8000,0020)
READ(8000,FFFF,1000)

WRITE – This command executes an MDIO write on the given register address. This command has one optional argument. If the optional <mask> argument is not given, then the WRITE command assumes a <mask> value of "FFFF" and the register at <register address> is written the exact value of <set or clear bits>. If all three arguments are given, the WRITE command only writes to the bits specified in the <mask> argument to the bit value of the corresponding bit in the <set or clear bits> argument.

Syntax: WRITE(<register address>, <set or clear bits>, [<mask>])
Example: WRITE(8000,FFFF)
WRITE(8000,F000,1001)

REPEAT – This command defines the start point of a loop. Loops can be nested and can exist within macros. The REPEAT command has a required argument <number of repeats>, which tells the software how many times to repeat a loop. The end of a loop is defined with a STOP command, which now makes it possible for a sequence to have multiple STOP commands. (See STOP definition for more details.)

Syntax: REPEAT <number of repeats>
Example: REPEAT 5 //Beginning of loop, run 5 times
DISPLAY(Looping Now…) //Commands within loop
STOP //Marks the end of a loop

ECHO – This command sets or clears a flag in the software. The flag determines if a WRITE command will print an output to the output textbox. If the flag is set, then the WRITE command will report an output. If the flag is cleared, there is no report of WRITE commands. To set the flag, use the ECHO ON command. To clear the flag, use the ECHO OFF command. The scope of the ECHO command is limited to the current sequence, and the default value for every sequence is ECHO ON. Therefore, if the user does not want any reports of WRITE commands, and is calling several macros, then the user needs an ECHO OFF command at the top of each macro.

Syntax: ECHO ON
ECHO OFF
Example: START
ECHO OFF //Clears the echo flag for this sequence
WRITE(8000,abcd)
STOP //End of sequence

SETDEVADD – The user can now change the device address using the SETDEVADD() command instead of using the settings menu. The current device address is displayed in the status bar. The command has a required argument called <devadd>. This argument is assumed to be a hexadecimal value, unless the user uses base-casting (for example: d'10 is decimal 10 and will be converted to Hex A, or H'A).

```
Syntax:    SETDEVADD(<devadd>)
Example:   SETDEVADD(01)      //Sets the device address to H'01 [PMA/PMD]
           SETDEVADD(d'30)    //Sets the device address to H'1E [Vendor Specific 1]
           SETDEVADD(b'100)   //Sets the device address to H'04 [PHY XS]
```

SETPHYADD – The user can now change the physical address using the "SETPHYADD()" command instead of using the settings menu. The current physical address is displayed in the status bar. The command has a required argument called <phyadd>. This argument is assumed to be a decimal value, unless the user uses base-casting (for example: h'10 is hexidecimal 10 and would not need to be converted before sending).

```
Syntax:    SETPHYADD(<phyadd>)
Example:   SETPHYADD(01)      //Sets the device address to H'01
           SETPHYADD(h'1E)    //Sets the device address to H'1E
           SETPHYADD(30)      //Sets the device address to H'1E
```

PAUSE – This command freezes execution of the sequence for the specified number of milliseconds. The maximum value is 65535 ms (slightly longer than 1 minute and 5 seconds), and the minimum value is 1 ms.

```
Syntax:    PAUSE(100)
Example:   START
               CLAUSE 22 //Sets the MDIO format to Clause 22
               WRITE(????,abcd)
               PAUSE(10) //Pauses execution for 10 ms
               CLAUSE 45 //Sets the MDIO format back to Clause 45
               READ(8000)
           STOP //End of sequence
```

CLAUSE – This command sets the MDIO format to either IEEE 802.3 Clause 22 or Clause 45, depending on the argument given. The argument can be either 22 or 45. The scope of the CLAUSE command is limited to the current sequence, and the default value for every sequence is Clause 45. So for a Clause 22 device, every sequence that is executed would need a CLAUSE 22 command immediately after the START command.

```
Syntax:    CLAUSE 22
           CLAUSE 45
Example:   START
               CLAUSE 22 //Sets the MDIO format to Clause 22
               WRITE(????,abcd)
               CLAUSE 45 //Sets the MDIO format back to Clause 45
               READ(8000)
           STOP //End of sequence
```

3 User Interface

This section describes the user interface and explains how to use its features. Figure 1 shows a screen shot of the GUI after starting the program.

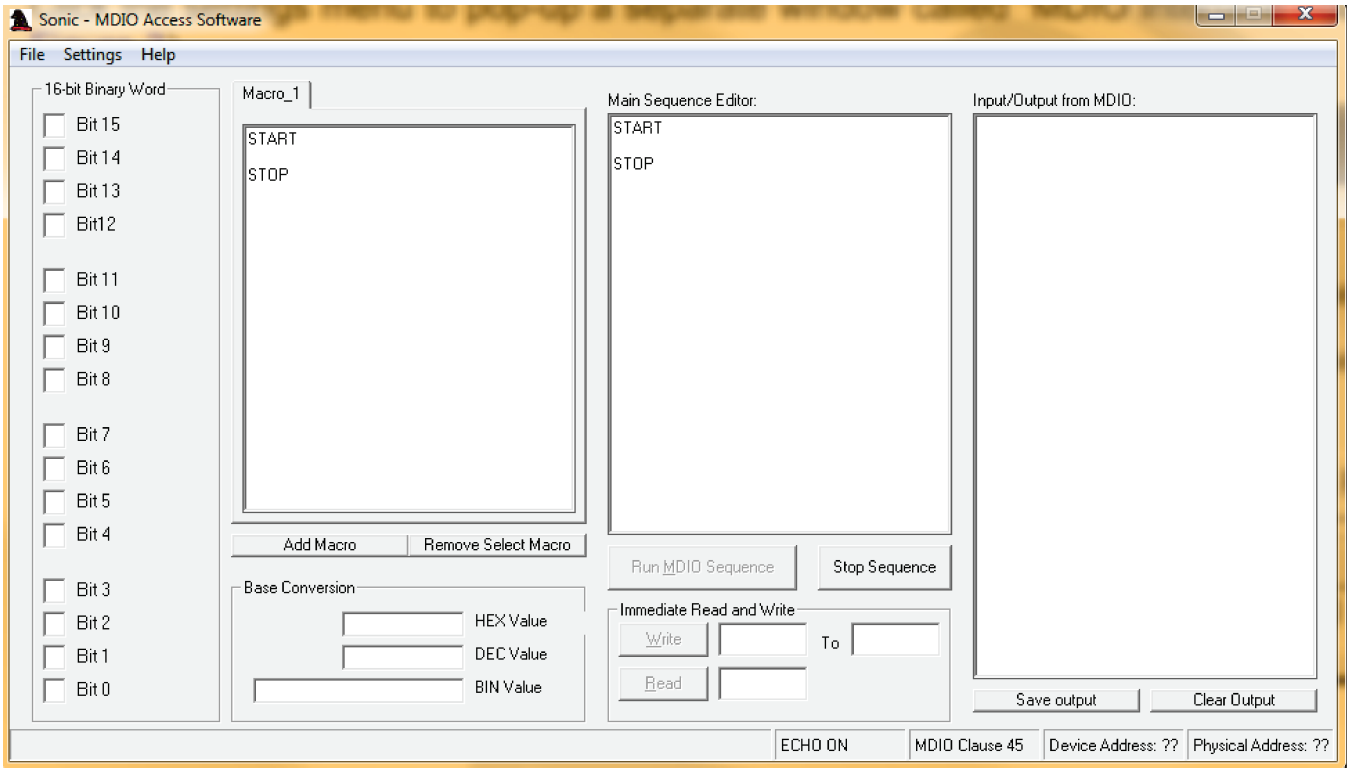

Figure 1. Main Screen

3.1 Main Sequence Editor

The “Main Sequence Editor” textbox is where the main sequence is typed. The *Run MDIO Sequence* button is below the Main Sequence Editor textbox. When this button is clicked, the software executes the sequence. The button to the right is labeled *Stop Sequence*. This button allows the user to abort the execution of a sequence at anytime.

3.2 Input and Output from MDIO

Output from any command during execution is printed into the “Input/Output from MDIO” textbox on the right side of the main sequence editor. The "Input/Output from MDIO" textbox is read-only. The buttons below this textbox allow the user to save the output to a file and clear the output. The user can also highlight and copy text from the textbox. Some commands have color-coded output to help quickly read the results. When the user saves the output, the user can choose standard text format or rich-text format. Saving in rich-text format keeps the color and font encoding.

3.3 Immediate Read and Write

The section below the main sequence editor, labeled “Immediate Read and Write,” can be used to issue a single READ or WRITE command. To read a register’s content, type the register address (in hexadecimal) in the textbox to the right of the READ command, and then press Enter or click the *Read* button. The register’s content will be printed to the output textbox. To write a value to a register, type the hexadecimal value into the textbox to the right of the *Write* button, and the address of the register (in hexadecimal) in the next textbox (to the right). Then, press Enter, or click the *Write* button. An output report will be printed to the output box.

3.4 Macro Tab Strip

Macros are stored on the left side of the main sequence editor under a tab with a textbox below. The two buttons below the tab allow the user to add new macros, or remove the currently selected macro. When the user has multiple macros defined and wants to select a different macro, left-click on the desired macro tab name to select it. If more macros are defined than fit across the top of the editor, a left and right arrow button automatically pops-up. Use the left and right arrow to scroll through the macro tabs. Right-click on any selected tab and a pop-up window appears (see Figure 2).

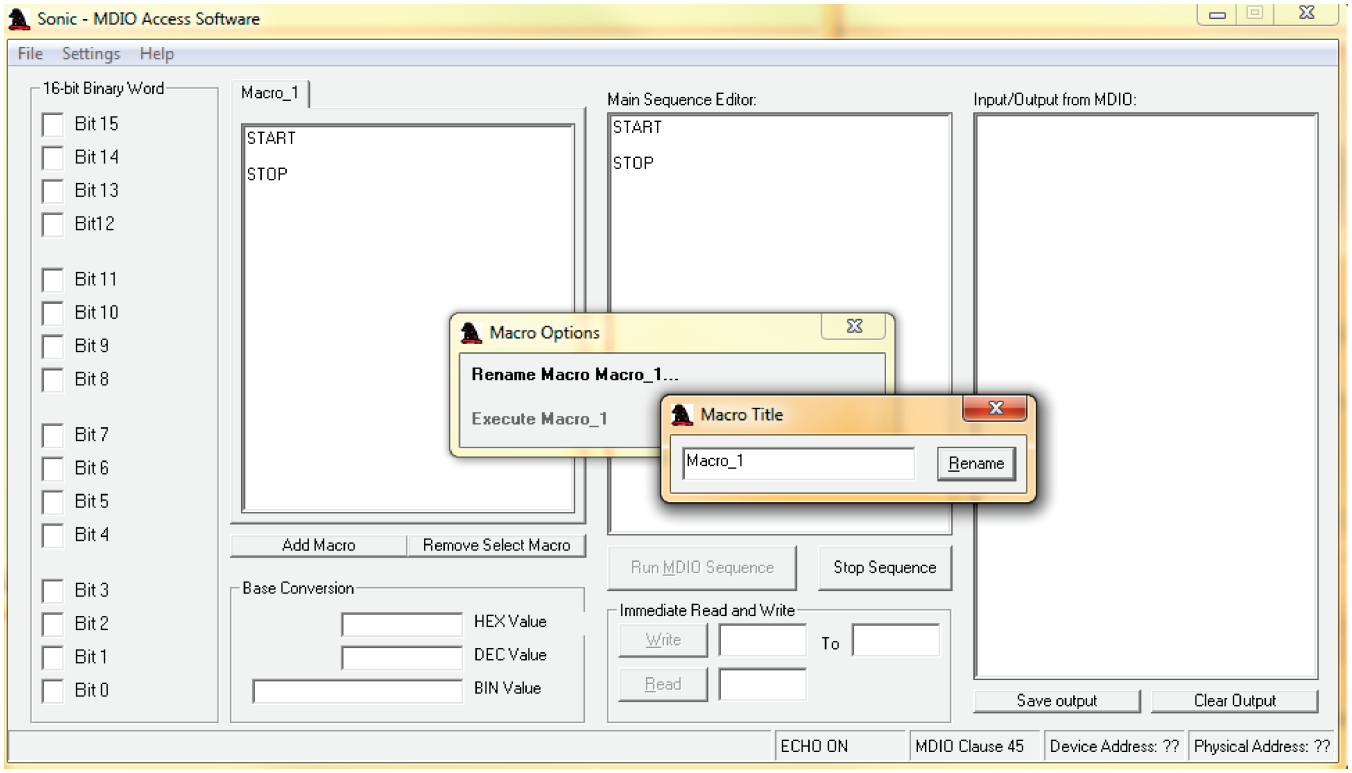

Figure 2. Macro Execution and Rename Pop-Up Windows

This window provides two options: rename the macro and execute the macro. Clicking on the rename option allows the user to type a new name for that macro. Clicking the execute option executes that single selected macro.

3.5 Status Bar

The status bar for the program is located at the bottom of the main program's window. There are four panels in the status bar. The first panel is for general information (for example: when running a sequence this panel shows "Executing Sequence…").

The second and third panel serve as indicators and toggle buttons. The second panel is for the ECHO command. It shows the current status of the echo flag at anytime. The echo status can be changed during sequence execution by either executing an ECHO OFF or ECHO ON command within the sequence, or it can be toggled at any point by clicking the panel in the status bar.

The third panel is for the MDIO clause mode. It can be clicked at any point to toggle the MDIO mode (Clause 45 or Clause 22), or it can be changed within the sequence with a CLAUSE 45 or CLAUSE 22 command.

The last panel shows the device address. This window updates anytime the user changes the device address, either by changing it in the settings menu, or by using the SETDEVADD() command in an MDIO sequence.

3.6 Base Conversion

The “Base Conversion” section is located below the macro tab strip. This section allows the user to type, or drag and drop, a valid numerical expression into the textbox labeled as the appropriate numerical base. It automatically converts the number to the other two listed numerical formats. It also populates the 16 textboxes labeled as “Bits[0-15]” on the left side of the window with the binary value for each bit.

If the user drags (or types) a hexadecimal value of “ABCD” into the “HEX Value” textbox, the program will report the decimal value (43981), and the binary value (1010101111001101) in the textboxes below.

If the user drags a value over a textbox (hexadecimal, decimal, or binary), and it is not a valid expression for that base, the program will not allow the user to drop the value into the textbox.

3.7 File Menu

The file menu allows the user to open, save, and close sequence files and exit the program. A sequence file stores the main sequence and all defined macros. The files are saved in plain text format and can be edited with any text editor. The format of the text file is explained in Section 2.

3.8 Settings Menu

Click the settings menu to pop-up a separate window called “MDIO Interface Board Settings…” (see Figure 3).

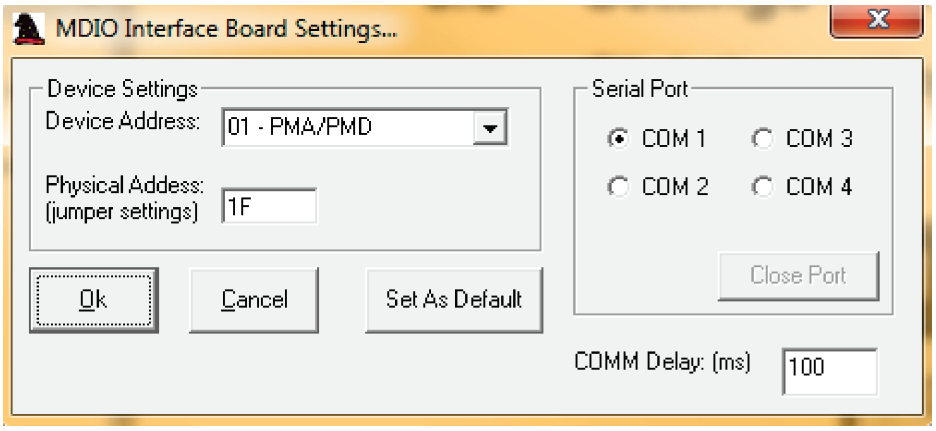

Figure 3. MDIO Interface Board Settings Menu

The settings menu defines both the device and serial port settings for the PC running this program. This software requires an MDIO interface board to communicate with the device. The program communicates with the board through a serial port. The board translates the serial port data into an MDIO datastream to send to the device. When data comes from the device, the process works in reverse.

The *Serial Port* radio buttons let the user select which serial port is used to communicate with the MDIO interface board.

The *Close Port* button allows the user to close an open serial port without having to exit the program. This is useful when the user wants to use another program at the same time that uses the same serial port.

The “COMM delay” setting controls a delay value that is used by the program between communication commands. The minimum value is 1 ms, and the maximum value 10000 ms.

The “device address” setting is a device value that is defined in the device design and determined by the device’s function according to IEEE 802.3. The default value is “1E”, which is a vendor-specific address. “04” would be used for a PHY device.

The “physical address” setting is a device value that is typically programmable to allow multiple devices on a single MDIO chain. On an EVM board, this address is usually programmable through jumpers on the board.

Before the user can execute a sequence or an immediate READ or WRITE, the user must select the serial port to use to communicate with the MDIO interface board. Click the settings menu and select the serial port, set the physical and device addresses, and click OK.

3.9 Help Menu

The help menu tells the user what version of the software they are using. It also displays the readme.txt file that explains new features and known issues. During installation of the software, the readme.txt file is saved in the same directory as the program.

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		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265