Interfacing TPS65381 With Hercules™ Microcontrollers

Haixiao Weng *MCU Safety Application*

ABSTRACT

The application report provides help to design a safety system with both the Hercules safety microcontrollers and the TPS65381 power supply. The document explains the hardware considerations and the software flowchart.

Project collateral and source code discussed in this application report can be downloaded from the following URL: http://www.ti.com/lit/zip/spna176.

Contents

List of Figures

1 Introduction

The Hercules safety microcontroller platform consists of three ARM® Cortex™-based microcontroller families: TMS470M, TMS570 and RM4x. Designed specifically for IEC 61508 and ISO 26262 safety critical applications, the Hercules platform provides advanced integrated safety features while delivering scalable performance, connectivity, and memory options.

The TPS65381 is a multi-rail power supply designed to supply microcontrollers in safety critical applications, such as those found in automotive.

This application report discusses how to integrate the Hercules MCUs and the TPS65381 together. The hardware and software examples apply to the RM46 control card.

2 Hardware Interface

Figure 1 shows the block diagram and connections between TPS65381 and a Hercules MCU; it is copied from the *TPS65381-Q1 Multi-Rail Power Supply for Microcontrollers in Safety-Critical Applications Data Sheet* (SLVSBC4). The RM46 control card design followed this recommendation with the following differences and highlights:

- In Figure 1, the SPI interface from the MCU side is MIBSPI1, 3, 5 while the control card uses SPI2 to talk to TPS65381.

  TPS65381 can accept both the SPI compatibility and MibSPI modes. In the RM46 control card, MibSP1 is reserved for the mother board, MibSPI3 is used for communication between the MCU and bridge driver, and MibSPI5 is multiplexed with Ethernet. So, SPI2 is assigned to talk to TPS65381.

  If MibSPI was used, the CPU load (RM46) could be reduced by combining several SPI transfers into one MibSPI transfer. For example, those four Q and A Watchdog answers can be combined into one MibSPI transfer. Note that the default SPICS inactive time between two adjacent 16-bit data (inside one MibSPI transfer frame) is much shorter than the SPICS inactive time that the TPS65381 can handle.

- In Figure 1, VCCAD is connected to 5 V while the control card uses 3.3 V as VCCAD input.

  This setup depends on your system. In this system, since the ADC input from the DRV8301 motherboard is trimmed in the range of [0, 3.3 V], a 3.3 V VCCAD and VREFHI can achieve the best dynamic range and accuracy.

- In the control card, a 10 nF capacitor is added to the DIAG_OUT.

  A cap of 10 nF to 33 nF is required, otherwise, the analog voltages on DIAG_OUT cannot be properly measured.

- In the control card, a DIP switch is placed between the TPS65381 (Pin NRES, ENDRV) and the Hercules device (Pin nPORRST, GIOA4) so that the user can choose ON (connect) or OFF (disconnect).

  This DIP switch is for debug purpose only. If the watchdog in TPS65381 is not served correctly, the TPS65381 might keep resetting the MCU. With this DIP switch, during debug stage, you can disconnect the NRES and MCU nPORRST if this saturation occurs.

  In production, this DIP switch is not necessary since the correct Flash image has been programmed into the device. If reprogramming a system after production is required, the application bootloader can force the TPS65381 to stay in diagnostic mode, re-program the Flash, and go to active mode after the program completes.

- In the control card, a 500Ω external pull up resistor is added to the ENDRV pin (TPS65381).

  The TPS65381 ENDRV pin internal pull-up resistor is 4.87kΩ. You must consider the pull down resistance on the high or low end driver side and the timing requirement of the bridge driver. If this 4.87 kΩ cannot pull up the circuit fast enough, a buffer or an external pull up resistor is required.

  The RM46 control card works with the DRV8301 motherboard, which has a 1kΩ pulldown resistor at this pin. The 500 Ω resistor on the control card is implemented to conqueror this 1kΩ pulldown resistor. Actually, removing the pulldown resistor on the DRV8301 board is preferred from the technical viewpoint, but it cannot be changed.

- There is a barrel connector in the RM46 control card while all the C2000™ control cards get power from the mother board (for example, DRV8301).

  The TPS65381 requires 5.8 V minimum supply while the DRV8301 motherboard can only provide 5 V. Therefore, power has to be provided separately.

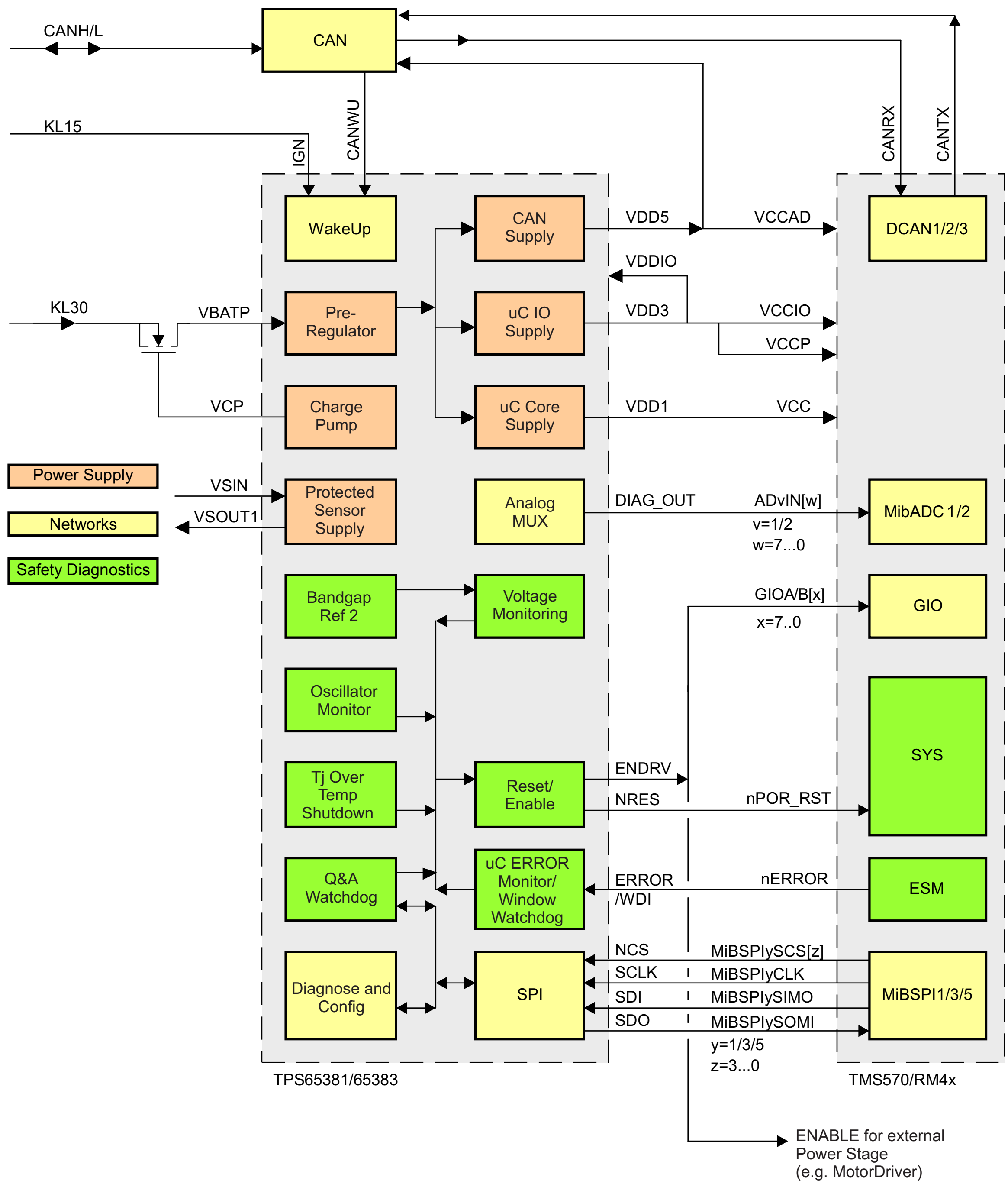

Figure 1. Example TPS65381 With Texas Instruments TMS570/RM4x

- In the control card, the following circuit is added to simulate an ignition pulse. The IGN input of TPS65381 should be kept low before the power supply (VBAT, TPS_POW in Figure 2) reaches 5.8 V. Otherwise, you have to toggle the IGN pulse again to get the TPS65381 working correctly. There is internal delay and deglitch in the IGN pin. However, if the power supply ramp is very slow (300 ms rising time from 0 V to 5.8 V), this internal delay is not long enough and a startup reset circuit is necessary (see Figure 2). Otherwise, the NRST could be released before the VBAT reaches 5.8 V and cause a TPS65381 BIST failure. To make it simple, this circuit can be removed if:
  – The power supply ramp time is fast (for example, 30 ms from 0 V to 6 V); Or
  – The application can apply an ignition pulse later on

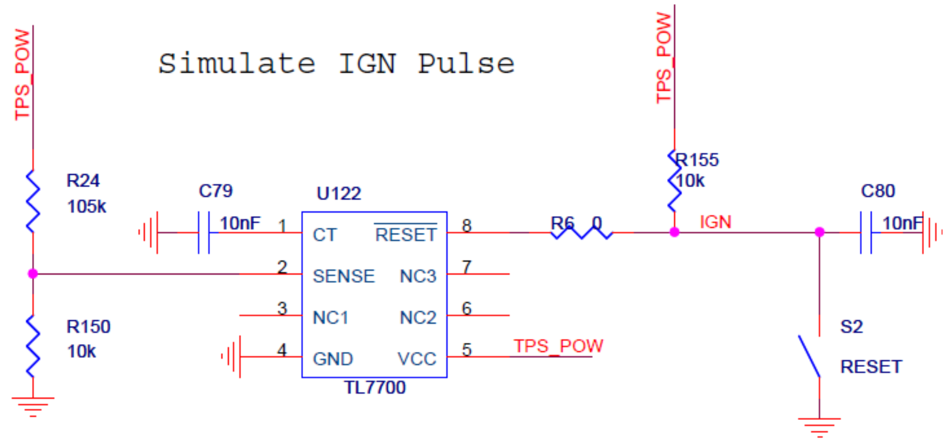

Figure 2. Circuit to Simulate the IGN Pulse

3 Software Flowchart

This section provides step-by-step notes on how to configure the TPS65381 with Hercules devices.

1. Initialize the SPI module.

   Make sure that the TPS65381 maximum speed for SPICLK is 5 MHz (3.3 V system) and the minimum setup time NCS (t_{sucs}) and hold time NCS (t_{hcs}) are 45 ns. The SPI module default settings (after reset) do NOT meet this requirement. Notice that SPI Transfer Inactive time (t_{hlcs}) should be at least 2-3 *sys. clock cycle* (*typ. 250 ns*). The application code has to set the wdelay field in the SPI Data Format Register (SPIFMT0) in the MCU side to meet this 750 ns requirement.

   The attached examples call the *spiInit()* function to initialize SPI2.

```
spiInit();
```

2. Force the TPS65381 to stay in Diagnostic State.

   The attached examples call *ecmpInit ()* to force the TPS65381 to stay in Diagnostic State. Inside this function, the Hercules devices set the TPS65381 *SAFETY_CHECK_CTRL* register to 0x17.

```
ecmpIfSetRegister(ECMP_SAFETY_CHECK_CONTROL, 0x17);
```

3. Configure the TPS65381 into Q and A watchdog mode and Hercules Error Pin mode.

   The following code inside *ecmpInit ()* configures the TPS65381 into Q and A mode and Hercules Error Pin mode.

```
ecmpIfSetRegister(ECMP_SAFETY_FUNCTION_CONFIG, 0xE4);
```

4. Perform Diagnostic test, for example, driving DIAG_OUT pin to the MCU's ADC module.

   The attached examples drive Voltage-monitor band gap (VMON_BG) to RM46 ADIN0 pin to check if it is in the range [2.2 V, 2.8 V].

```
ecmpDiagMuxOutputMode(2); //analog output
ecmpDiagMuxSelectSignal(0x80);//select VMON Bandgap
ecmpDiagMuxEnable();
```

```
for(temp=0;temp<20000;temp++);//add some delay so that the output is stable
adcInit();
adcStartConversion(adcREG1,1);
while(adcIsConversionComplete(adcREG1,1)==0);
adcGetData(adcREG1,1, ADC_Group1);
if((ADC_Group1[0].value <2731) || (ADC_Group1[0].value > 3475))
{
//VMON_BG is out of [2.2v 2.8v]
sciSend(scilinREG, 28, (uint8_t *) "\r\nVMON_BG is out of range!\r\n");
while(1);
}

ecmpDiagMuxDisable();
```

5. Force the program into an infinite loop.

   This step is for debug purpose ONLY and should be removed in production. During debug, you can connect to device using Code Composer Studio™ (or other IDE) and move the program counter (PC) to next instruction to run the program.

```
asm(“ b #-8”)
```

6. Synchronize the MCU Watchdog triggers on the Watchdog timer.

   Before serving the watchdog, the application code must synchronize the MCU watchdog triggers and the watchdog timer (in TPS65381). The attached example start to serve the open window after configuring the watchdog window because *"any write access to WDT_WIN1_CFG or WDT_WIN2_CFG registers immediately starts a new Watchdog Token-Response Sequence Run"* (SLVSBC4). An alternative way to synchronize is: "Don't serve the watchdog until the watchdog timeout". (The Q and A open window starts after watchdog timeout).

7. Serve the watchdog.

   (a) Watchdog answer calculation.

   The function *ecmpSendWdgAnswer(void)* reads the watchdog token and watchdog status register to calculate the watchdog answer. The algorithm is shown in the *Watchdog Token Value Response Calculation (or Watchdog Answer Calculation)* figure in the *TPS65381-Q1 Multi-Rail Power Supply for Microcontrollers in Safety-Critical Applications Data Sheet* (SLVSBC4).

```
void ecmpSendWdgAnswer(void)
{
    unsigned short answer;
    unsigned short token0  = ecmpIfGetRegister(ECMP_WDG_TOKEN) & 0xF;
    unsigned short token1  = (token0 >> 1) & 1;
    unsigned short token2  = (token0 >> 2) & 1;
    unsigned short token3  = (token0 >> 3);
    unsigned short aswCnt0 = (ecmpIfGetRegister(ECMP_WDG_STATUS) >> 6) & 3;
    unsigned short aswCnt1 = (aswCnt0 >> 1) & 1;

    token0  &= 1;
    aswCnt0 &= 1;

    answer =  (token0 ^ (aswCnt1 ^ token3))                  |
             ((token0 ^ aswCnt1  ^ (token1  ^ token2)) << 1) |
             ((token0 ^ aswCnt1  ^ (token3  ^ token1)) << 2) |
             ((token2 ^ aswCnt1  ^ (token0  ^ token3)) << 3) |
             ((token1 ^ aswCnt0)                       <, 4) |
             ((token3 ^ aswCnt0)                       << 5) |
             ((token0 ^ aswCnt0)                       << 6) |
             ((token2 ^ aswCnt0)                       << 7);

    ecmpIfSetRegister(ECMP_WDG_ANSWER, answer);
```

```
}
```

This function is calculating one answer and needs to be called 4 times in one watchdog period. After each answer generation the answer needs to be sent to the TPS65381 to decrement the answer counter. Then the function can be called to calculate the next answer. This function is only valid for WDT_TOKEN_FDBCK [7:4] = 0 (default condition).

(b) Trigger watchdog in MCU side.

To serve the watchdog within the correct window, the attached two examples are slightly different. In the project CCS_RM46_NoRTOS, the watchdog trigger is controlled by the RTI interrupt. The RTI interrupt service routine (ISR) calls the *ecmpSendWdgAnswer()* to serve the watchdog. The project CCS_RM46_FreeRTOS create a task vTaskSendWdgAnswer to serve the watchdog. The watchdog trigger is controlled by the vTaskDelay function.

In both cases, MCU serves the watchdog at similar time stamp as shown in Figure 2. It provides three open window watchdog responses in the middle of the open window and the fourth response 5 ms after the close window starts. As mentioned in the *TPS65381-Q1 Multi-Rail Power Supply for Microcontrollers in Safety-Critical Applications Data Sheet* (SLVSBC4), “After this final correct SPI WD Token response, next TOKEN will be generated within 1 sys. clock cycle (typ. 250ns), after which next WD OPEN WINDOW (Q and A + 1) starts”. Therefore, MCU and the TPS65381 watchdog are synchronized as the fourth response is provided.

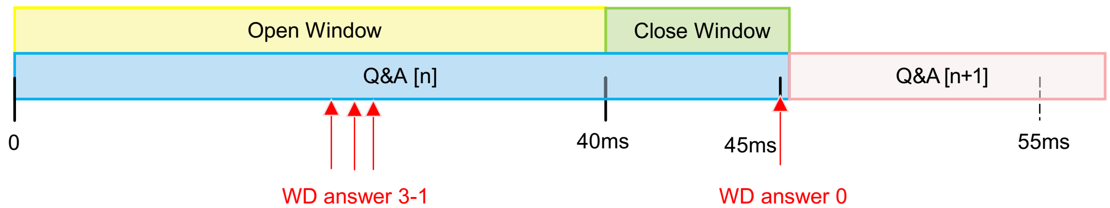

Figure 3. WDT Response Timing in the Examples

(c) Go to active state.

Once the watchdog failure counter reduces to zero, the example code.

(i) Enable the watchdog reset.

```
ecmpEnableWdg()
```

(ii) Force the MCU leaving the diagnostic mode.

```
ecmpLeaveDiagnosticState()
```

Besides forcing the TPS65381 leaving diagnostic state, this function cleans the error status register (If this register is not equal to zero, the TPS65381 will go to safe state) and enables the ENDRV signal.

4 Troubleshooting Hints

Q. My program cannot go to the diagnostic state. It stays at:

```
while (ecmpIfGetBit(ECMP_WDG_STATUS, 2) == 1);
```

A. Make sure the DIP switch PORRST position is ‘on’. If it is ‘off’, the MCU SPI might send the command before the TPS65381 is ready (it is running a selftest). After that, power cycle the board or press the PORCYC push button (S2) to generate a IGN pulse.

Q. After programing the RM46 device and click ‘run’ in CCS, it goes to the safe state.

A. Usually, after programing the Flash, you need to power cycle the board to get the code running correctly. TPS65381 cannot go back to diagnostic or active state from safe state without resetting the device. The recommended debug sequence is:

1) Program the Flash.

2) Make sure the DIP switch nPORRST position is ‘on’.

3) Power cycle the board.

4) Connect with Code Composer Studio. Now the code should stops at the:

```
asm(“ b #-8”)
```

5) Move the PC to next instruction.

6) Click ‘run’ in Code Composer Studio.

Q. The TPS device keeps resetting the MCU (the PRST orange LED is flashing), what can I do?

A. Switch off the PORRST DIP position. Or, hold on the nRST push button (S1), press or release the PORCYC push button (S2), after one second, release the nRST button. By doing this, the TPS655381 device will be forced into the safe state, in which the nPORRST is high, 3.3 V and 1.2 V are provided normally and ENDRV is driven low. Then, you can re-flash the device.

Q. My code works well without optimization. However, after I enabled the optimization, the TPS65381 watchdog fails and reset the MCU, what happened?

A. With optimization, the time between two adjacent SPI transfers is shorter. Check t_{hlcs}, the minimum CS high time between two adjacent SPI transfers. On the Hercules devices, this can be adjusted by the SPI Transmit Data Register (SPIDAT1) and the SPI Data Format Registers (SPIFMT[0-2]). It is recommended to set the wdelay field in the SPIFMT register as 0x3F (with 80 MHz VCLK) as a tradeoff between efficiency and timing margin. For more details, see Section 5.

5 Examples

Two examples are provided to demo the TPS65381 safety drivers based on the RM46 control card: a FreeRTOS project and a NoRTOS project (without any RTOS). The FreeRTOS and NoRTOS project support the HyperTerminal (19200 baud rate, two stop bits and no parity). This section briefly explains how all these three projects are generated.

5.1 FreeRTOS Project

1. Open the FreeRTOS_CCS_RM46.hcg with the HalCoGen 03.02.02, generating code. (The RM48 FreeRTOS template is used because HalCoGen 03.02.02 does not support RM46 FreeRTOS)
2. Copy os_app.c, ecmp_if.c and ecmp.c to source; Copy ecmp_if.h and ecmp.h to include;
3. Modify sys_main.c and notification.c. All the modifications are in-between /* USER CODE BEGIN */ and /* USER CODE END */.

5.2 NoRTOS Project

1. Open the NoRTOS_CCS_RM46.hcg with the HalCoGen 03.02.02, generating code.
2. Copy ecmp_if.c and ecmp.c to source; Copy ecmp_if.h and ecmp.h to include;
3. Modify sys_main.c and notification.c. All the modifications can be tracked by the /* USER CODE BEGIN */ and /* USER CODE END */.

6 References

- *RM46x 16/32-Bit RISC Flash Microcontroller Technical Reference Manual* (SPNU514)
- *RM46L852 16/32-Bit RISC Flash Microcontroller Data Sheet* (SPNS185)
- *TPS65381-Q1 Multi-Rail Power Supply for Microcontrollers in Safety-Critical Applications Data Sheet* (SLVSBC4)

Appendix A Revision History

This document has been revised from SPNA176 to SPNA176A because of the following technical change(s).

Table 1. SPNA176A Revisions

Location	Additions, Deletes, and Edits
Figure 1	Remove [z] after SIMO and SOMI.

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