Application Report
SLAA334B–September 2006–Revised August 2018

MSP430 Flash Memory Characteristics

MSP430 Applications

ABSTRACT

Flash memory is a widely used, reliable, and flexible nonvolatile memory to store software code and data in a microcontroller. Failing to handle the flash according to data-sheet specifications can result in unreliable operation of the application. This application report explains the physics behind these specifications and also gives recommendations for the correct management of flash memory on MSP430™ microcontrollers (MCUs). All examples are based on the flash memory used in the MSP430F1xx, MSP430F2xx, and MSP430F4xx microcontroller families.

Contents

1 Flash Memory 2
2 Simplified Flash Memory Cell 2
3 Flash Memory Parameters 3
3.1 Data Retention 3
3.2 Flash Endurance 5
3.3 Cumulative Program Time 5
4 Flash Enhancements With Software 7
4.1 EEPROM Emulation With Flash 7
4.2 Enhancing Flash Data Retention Time With Flash Refresh 7
4.3 Verify Flash Data With a Checksum or CRC 8
5 Conclusion 8
6 References 8

List of Figures

1 Flash Memory Cell 2
2 Erasure of Flash Memory Cell (Erased Cell Is on the Right) 2
3 Programming a Flash Memory Cell (Programmed Cell Is on the Right) 3
4 Leak Mechanism in Tunnel Oxides 4

Trademarks

1 Flash Memory

Flash memory is one of the most popular nonvolatile memories to store program code and constant data values. Many microcontrollers, such as the MSP430 family of microcontrollers, have integrated flash memory for nonvolatile data storage. But there are big differences in behavior and performance of flash memory cells. This report explains the parameters and behavior of the ultra-low-power flash memory used in the MSP430F1xx, MSP430F2xx, and MSP430F4xx families.

2 Simplified Flash Memory Cell

A flash memory cell is based on a transistor with a floating gate. Figure 1 shows a simplified schematic of a flash memory cell. The ultra-low-power flash memory cell of a MSP430 is slightly modified, but the basic function is the same.

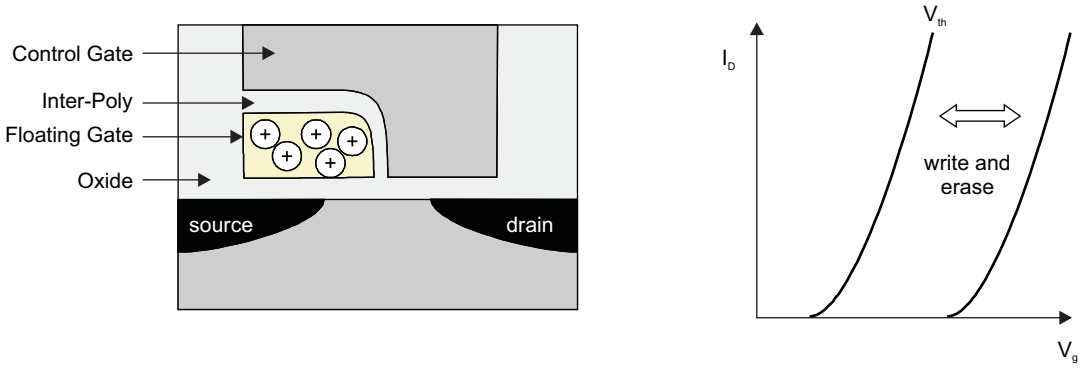

Figure 1. Flash Memory Cell

Underneath the control gate is an additional floating gate. This floating gate is either fully charged or discharged. Charging and discharging this floating gate is done with high energy through the oxide. The amount of charge in the floating gate influences the threshold of the transistor, which generates a logical 1 or 0 when this flash cell is read.

Erasing a flash cell, as shown in Figure 2, is positive charging the floating gate. A positive-charged floating gate results in reading a logical 1 from this erased memory cell. Negatively charging the floating cell, as shown in Figure 3, means programming this cell. The CPU reads from a programmed cell the logical value 0. For erasing and programming, high energy is needed to transport the charge through the oxide by FN tunneling for erasure or channel hot electron (CHE) injection for programming.

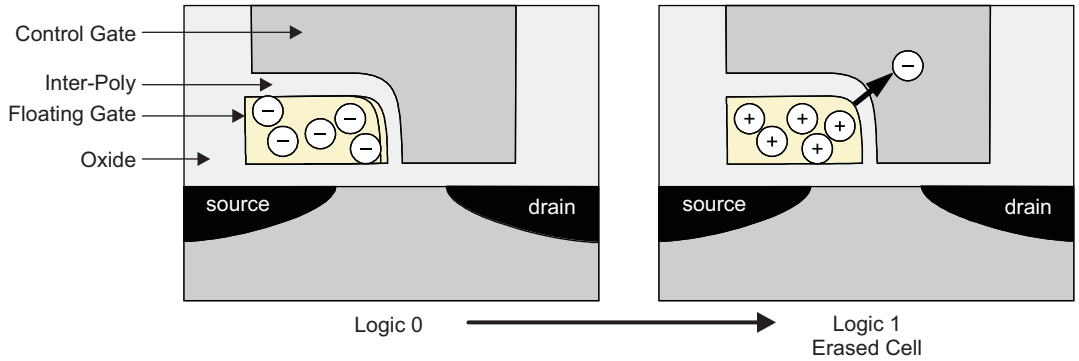

Figure 2. Erasure of Flash Memory Cell (Erased Cell Is on the Right)

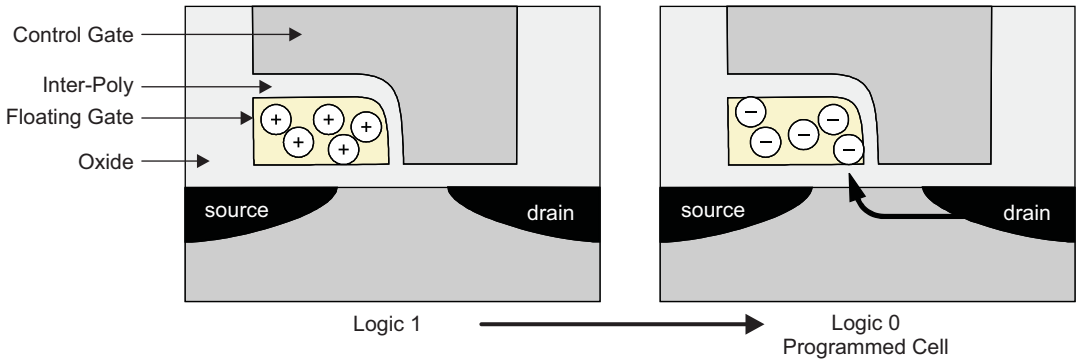

Figure 3. Programming a Flash Memory Cell (Programmed Cell Is on the Right)

The smallest unit that can be erased in a flash memory is a complete flash segment. Erasing flash means generating logical 1s in the memory. The MSP430 main flash memory has a segment size of 512 bytes. All MSP430 devices also have some smaller 64-byte or 128-byte flash segments. This area of the flash is called the information memory. In most applications, the main memory stores program code and constant data values, and the information memory is used for calibration data, serial numbers, etc. Except for the segment size, there is no difference between the main and the information memory. Hence, both can store program code or constant data values or both.

Programming flash memory generates logical 0 values in the flash memory cell. Programming can be performed bit, byte, or word wise. Flash segment size has no influence on programming of flash cells.

Only logical 0 can be written as single bits, bytes, or words. Because writing logical 1 in flash memory is only possible with erasing the memory, 1s can only be written segment-wise.

A high voltage is needed to program and erase flash memory. All MSP430 devices have an integrated charge pump and automatically generate this high voltage, when necessary. Also, a flash timing generator for automatic generation of erase and programming sequences is integrated in every MSP430. Hence, MSP430 flash programming is done without any external components. For programming and erasure, it only is important to keep the supply voltage V_{CC} within the data sheet limits and to program the input clock frequency of the flash timing generator according to data sheet requirements.

3 Flash Memory Parameters

3.1 Data Retention

3.1.1 Leakage Mechanism

Data retention is limited by leakage current through the insulating oxide. Leakage can only occur if the floating gate is fully charged. Therefore, leakage only can flip an erased cell with the logic level 1 to a programmed cell with the logic level 0. According to Manabe [1], there are several phenomena that cause leakage.

- Conventional stress induced leakage current (SILC) (see Figure 4a) explained by single trap-assisted tunneling conduction
- Anomalous SILC explained by leakage path, hopping conduction mechanism (HCM) large leak, little E-field dependence, intermittent (see Figure 4b)
- Trapped charges can block up the leak (see Figure 4c).

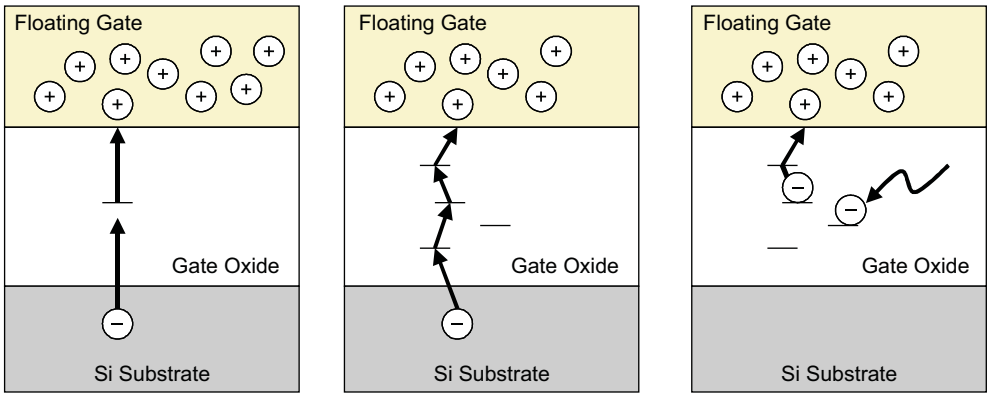

Figure 4. Leak Mechanism in Tunnel Oxides

Leakage behavior depends on temperature, program/erase cycles, lifetime, and process. In an end application, the main influence on data retention is temperature.

3.1.2 Data Retention Time

A common concern with nonvolatile erasable memories is data retention. As explained in Section 3.1.1, over time, the floating gate charge is reduced due to leakage introduced by oxide defects. With higher temperatures, leakage current increases and, thus, the charge on the floating gate is reduced more quickly than at lower temperatures. This temperatures dependence follows the Arrhenius equation (see Equation 1):

$$AF = e^{-\frac{Ea}{k}\left(\frac{1}{T1}-\frac{1}{T2}\right)} \quad (1)$$

where

- AF = Acceleration factor
- Ea = 0.6 eV = Activation energy
- k = 86.17 × 10^{-6} = Speed constant
- T1 = Temperature 1 (K)
- T2 = Temperature 2 (K)

The Arrhenius equation gives an acceleration factor (AF) for data retention based on a temperature difference. Because it would take too long to measure a data-retention time at 25°C, these measurements are done at a much higher temperature to accelerate the process. In the data sheets of MSP430 devices, data retention is specified to exceed 100 years at 25°C (see Table 1). This value is industry accepted and all vendors specify 100 years as the flash data retention duration at 25°C, despite it being extremely conservative.

Table 1. Data Retention Over Recommended Operating Temperature in Data Sheets

PARAMETER		TEST CONDITIONS	MIN	MAX	UNIT
$t_{Retention}$	Data retention duration	T_J = 25°C	100		years

Further tests show the exact number of years at 25°C is 1324 years.[6]

Equation 2 is an example of using the Arrhenius equation to calculate data retention for any ambient temperature:

T1 = 50°C = 323K
T2 = 25°C = 298K (data-sheet specification)

$$AF = e^{-\frac{Ea}{k}\left(\frac{1}{T1}-\frac{1}{T2}\right)} = e^{-\frac{0.6eV}{86.17\times10^{-6}}\left(\frac{1}{323K}-\frac{1}{298K}\right)} \approx 6 \quad (2)$$

Stored data ages six times faster at 50°C than it ages at 25°C.

In an application operating at 50°C, 24 hours per day, 7 days per week, the data retention time (specified in Understanding MSP430 Flash Data Retention) is reduced from 1324 years to 1324/6 ≈ 220 years.

If the application is running 5 hours per day at 50°C and the rest of the day at 25°C, the data retention is calculated as shown here:

5 hours/day at 50°C is equivalent to data aging at 25°C for 5 hours × 6 = 30 hours

19 hours/day at 25°C is equivalent to data aging at 25°C for 19 hours

Total data aging per day is 30 hours + 19 hours = 49 hours ≈ 2 days at 25°C

In this scenario, data retention is 662 years.

Erased MSP430 flash memory bits have fully charged floating gates and read as logical 1s. Therefore, single-bit failures due to leakage currents show erroneous 0s that should be 1s. These single-bit failures based on leakage currents can never read an erroneous 1, instead of expected 0s.

3.2 Flash Endurance

Like any other erasable memory, flash devices have a limited number of erase and write cycles they can withstand without failure. The reason for the limitation depends on either charge trapping characteristics or the dielectric breakdown characteristics of the tunnel oxide. This introduces a term called endurance. Endurance is a measure of the number of erase and write cycles that a flash array can achieve while retaining data integrity. According to *IEEE Standard Definitions and Characterization of Floating Gate Semiconductor Arrays* [2], endurance is defined as "The measure of the ability of a nonvolatile memory device to meet its data-sheet specification as a function of accumulated nonvolatile data changes." Table 2 lists the flash endurance specified in MSP430 data sheets.

Table 2. Flash Endurance in the Data Sheets

PARAMETER	MIN	NOM	MAX	UNIT
Program and erase endurance	10^4	10^5		cycles

The data sheet specifies a minimum of 10000 read and write cycles, while the devices typically fulfill 100000 read and write cycles.

MSP430 endurance testing shows that most devices easily achieve more than 100000 cycles at ambient or high temperature, while the number of cycles at low temperature is in the range of a few ten thousands. Failures are always a single bit failing erase. Measurements show improvement in the number of cycles if a quiescent period after erase operations is added.

The most common problem is stuck cells caused by charge trapping. Charge trapping occurs in the insulating tunnel oxide during erase operations, causing the cell to read a logic 0 although erased. This is a self-healing effect and usually detraps automatically in the quiescent period after an erase cycle. During endurance testing at Texas Instruments, the flash cells are continuously erased and rewritten. With a delay of at least one or two seconds between two erase and write cycles, the flash endurance increases significantly during the tests.

3.3 Cumulative Program Time

It is very easy to erase and program flash memory of a MSP430 with user-written software. A chapter explaining how to do this is in the family user's guides, and code examples for flash erasure and programming are available at http://www.ti.com/msp430.

It is possible to program single bits, bytes, or words of MSP430 flash memory. The designer can also program any 16-bit word up to two times and, with this method, can change additional bits from logic 1 to 0. But the cumulative program time parameter limits the number of over-programming operations between two erase cycles.

Before programming flash by software, the input clock divider of the flash timing generator must be programmed to a correct value. The data sheet allows a flash timing generator frequency between 257 kHz and 476 kHz. Please check the data sheet for correct values for each MSP430 derivative.

The MSP430F1xx and MSP430F4xx data sheets also state that programming one flash cell (byte or word) needs 35 flash timing generator clock cycles. During these 35 clock cycles, the high voltage for flash programming internally is applied for only 29 clock cycles, 6 clock cycles less than the complete cycle.

The MSP430F2xx data sheets specify flash programming a higher speed, as only 30 clock cycles to program one flash cell are needed. For flash programming, the high voltage is only applied for 27 cycles, 3 cycles less than the complete programming cycle.

Each time a single bit, byte, or word is programmed, a complete row of 64-byte flash cells sees the high voltage necessary for programming. This high voltage generates some stress to the complete row of flash cells, and this stress must be time limited to avoid damage. The next erase cycle resets this stress time to zero, and the cumulative program time restarts again from the beginning. According to the data sheet, as shown in Table 3, this high-voltage stress must be limited to 10 ms between two erase cycles. See the data sheets for the correct values for each MSP430 derivative.

The same 16-bit flash word cannot be programmed more than twice before the next erase cycle. Writing to one 16-bit word with two byte-wise programming cycles counts as two programming cycles. Single-bit overprogramming is possible only once, if the flash cell previously has been programmed 16-bit word wise.

Writing to the same row too many times can result in write disturb, and erased bits will be programmed as well. This produces no physical damage and, after erase, the disturbed bits are programmable as before. No long term effects are known.

Table 3. Example of Cumulative Program Time in the Data Sheets

PARAMETER	TEST CONDITIONS	V_{CC}	MIN	MAX	UNIT
Cumulative program time	(1)	2.7 V, 3.6 V		10	ms

(1) The cumulative program time must not be exceeded when writing to a 64-byte flash block. This parameter applies to all programming methods: individual word write, byte write, and block write.

The following examples show some scenarios and the influence of the cumulative program time on possible over-programming of one cell.

Example 1

The flash timing generator is programmed to its minimum frequency of 257 kHz. Under this condition, the high-voltage stress to program one single flash byte or word lasts for:

$$t_{high\_voltage} = \frac{1}{257\ kHz} \times 29 = 113\ \mu s \quad (3)$$

This allows, within the cumulative program time of 10 ms/113 µs = 88 programming cycles for bits, bytes, or words. To program one flash row of 32 words (64 bytes) word wise, 32 programming cycles are necessary. After these 32 programming cycles, in this example, additional 56 programming cycles are left for programming additional bits of this flash row. Byte-wise programming needs 64 programming cycles for one flash row, and this leaves only 24 programming cycles for additional bit programming.

Example 2

The flash timing generator is programmed to its maximum frequency of 476 kHz. Under this condition, the high-voltage stress to program one single flash byte or word is:

$$t_{high\_voltage} = \frac{1}{476\ kHz} \times 29 = 61\ \mu s \quad (4)$$

In this case, there is 10 ms/61 µs = 164 cycles for bit, byte, or word programming available, before the flash segment must be erased. Programming the 32 words of this flash line byte wise uses 64 cycles of the available 164 cycles. Hence, theoretically, 100 additional flash programming cycles are left to add additional logic 0 levels to this flash row. Because programming each word more than two times is not allowed, only another 64 cycles can be executed.

Programming flash in the block write mode is quicker and, therefore, leaves more time for single-bit over-programming operations within one flash row.

When using the digitally controlled oscillator (DCO) as clock source for the flash timing generator, it is important to check the worse-case accuracy of the DCO frequency. In many MSP430 derivatives, the DCO frequency is very temperature dependent and also has some V_{CC} dependence. Therefore, it is good practice to select a flash timing generator frequency in the middle of the allowed range. To program the flash timing generator frequency close to one of its data-sheet limits, use an accurate clock source. Either a high-frequency crystal clock source or calibration of the DCO frequency using a 32-kHz crystal clock are options.

Programming the flash with a flash timing generator frequency outside the data-sheet specifications can give correct results at first but can lead to reduced data-retention time and affect flash reliability. Not following the cumulative programming time parameter can also result in bit failures and generally reduced reliability of the MSP430 flash memory.

4 Flash Enhancements With Software

4.1 EEPROM Emulation With Flash

The main MSP430 memory is divided into 512-byte flash segments. Depending on the family, MSP430 devices have either additional two segments with 128 bytes each or four segments with 64 bytes each. These smaller segments are called information memory. Both information memory and main memory are functionally equal and can be used to store program code and data. However, in most applications, it is convenient to use the main memory for program code and static constant values. The small information memory segments allow storage of nonvolatile variables, which easily can be modified at run time. Besides the small segment size, the high number of erase and write cycles (high flash endurance) allow emulation of an EEPROM and often make an external EEPROM unnecessary.

4.2 Enhancing Flash Data Retention Time With Flash Refresh

As explained in Section 3.1.2, data retention time is very much dependent on the ambient temperature of the MSP430 application. One possible solution to enhance flash data retention is to refresh the flash contents from time to time with software.

In an ideal scenario, the application has idle time frames, where no external events must be observed. During such an idle time, the software can copy one flash segment into RAM or any other flash segment. After erasing the original segment, the content is copied into the original segment. After such a flash refresh cycle, the data retention time for this segment restarts.

Example: At 50°C, the data-retention time is reduced to 16 years. In this case, a flash refresh every 16 years expands the data-retention time for another 16 years. With 100000 erase and write cycles, the total data retention time in this scenario is expanded to 1.6 million years.

As mentioned in an previous section, the number of erase and write cycles (flash endurance) at higher temperatures is higher than at lower temperatures. Hence, the typical value of 100000 erase and write cycles can be used for the calculation here.

CAUTION

The application must ensure that flash refresh of a flash segment is not interrupted by a power failure.

Failing V_{CC} after erasure of the flash sector containing the interrupt vector table leads to a complete failure of the application. Capacitive decoupling must ensure that, during the flash refresh cycle, the current for erasure and programming does not drop V_{CC} below the required erasure and programming voltage as specified in the data sheet.

If a power failure aborts a flash segment erase of flash programming process, this flash segment can be erased and reprogrammed.

4.3 Verify Flash Data With a Checksum or CRC

In safety critical applications, it is good design practice to verify the flash contents from time to time with the application software. The simplest method is a checksum across the complete flash memory. The memory check can be done each time the application is turned on. If an application is running for a long time, a memory check can be done every day, week, or month. It is not the focus of this application report to discuss the various options of checksum or CRC checks. Refer to CRC Implementation with MSP430 MCUs for more information.

5 Conclusion

Flash memory is a widely used, reliable, and flexible nonvolatile memory to store software code and data in a microcontroller. However, as with any programmable and erasable memory, physical parameters must be understood, and the application must be designed accordingly to enable safe data storage over a long time. To ensure reliable operation of the application for a long time, data-retention time, flash endurance, and cumulative program time must be handled according to the data-sheet specifications. Occasionally checking the flash contents with a checksum or CRC enhances product reliability.

6 References

1. Y. Manabe, *Detailed Observation of Small Leak Current in Flash Memories With Thin Tunnel Oxides*, IEEE Std 1998, pp 95–99
2. *IEEE Standard Definitions and Characterization of Floating Gate Semiconductor Arrays*, IEEE Std 1005–1991, 17 Oct 1991
3. Michael Jonas and Hans-Martin Hilbig, *MSP430 Endurance Testing*, TI Technical Journal, April–June 2000
4. CRC Implementation with MSP430 MCUs
5. Sunil G. Gaitonde and Tenkasi V. Ramabadran, *A Tutorial on CRC Computations*, IEEE Micro, August 1988, pp 62–74
6. Understanding MSP430 Flash Data Retention

Revision History

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

Changes from April 14, 2008 to August 24, 2018	Page
• Editorial and format changes throughout document	1

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources 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.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

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