The following document contains information on Cypress products. The document has the series name, product name, and ordering part numbering with the prefix “MB”. However, Cypress will offer these products to new and existing customers with the series name, product name, and ordering part number with the prefix “CY”.

How to Check the Ordering Part Number

1. Go to www.cypress.com/pcn.
2. Enter the keyword (for example, ordering part number) in the **SEARCH PCNS** field and click **Apply**.
3. Click the corresponding title from the search results.
4. Download the Affected Parts List file, which has details of all changes

For More Information

Please contact your local sales office for additional information about Cypress products and solutions.

About Cypress

Cypress is the leader in advanced embedded system solutions for the world's most innovative automotive, industrial, smart home appliances, consumer electronics and medical products. Cypress' microcontrollers, analog ICs, wireless and USB-based connectivity solutions and reliable, high-performance memories help engineers design differentiated products and get them to market first. Cypress is committed to providing customers with the best support and development resources on the planet enabling them to disrupt markets by creating new product categories in record time. To learn more, go to www.cypress.com.

AN204730

F²MC-8FX Family 8-Bit Microcontroller MB95310/370 Series I²C Library Usage API

Associated Part Family: MB95310/370 Series

This document introduces API for I²C library.

Contents

1 Introduction....1
2 I²C Library Function List....1
3 I²C Function Detail....2
3.1 Initial_I²C Function....2
3.2 Write_I²C Function....2
3.3 Read_I²C Function....3
4 Usage Demo....4
4.1 Usage Steps of MB95F310 MCU....4
4.2 Software Usage Attention....6
4.3 Sequence of Data Transfer....6
5 Debugging....7
6 Additional Information....9
Document History....10

1 Introduction

This document introduces API for I²C library.

Cypress MCU MB95F310 has I²C module and it can be used by connecting master device to slave device. In following chapter we will describe the library of Cypress MCU MB95F310.

2 I²C Library Function List

This section introduces all functions in I²C library in project Simulate LCD EVBoard.prj which uses MB95F310 as MCU.

Table 1 lists the I²C functions.

Table 1. I²C Functions

Function name	Description
void initial_I²C(void)	Initializes I²C and sets shift clock frequency
void Write_I²C(unsigned char addr, unsigned char subadd, unsigned char data)	Writes data to slave chip
unsigned char Read_I²C(unsigned char addr, unsigned char subaddr)	Reads data from slave chip

3 I²C Function Detail

This section introduces the detail of I²C functions.

3.1 Initial_I²C Function

Table 2 describes initial_I²C function.

Table 2. AD_Init Function

Function name	Initial_I²C
Function prototype	void initial_I²C(void)
Behavior description	Initializes I²C and sets shift clock frequency
Input parameter	None
Return value	None
Example	Sets shift clock frequency to 23 KHz when MCLK is 6 MHz: initial_I²C();

This function is to initialize I2C. In this function default shift clock frequency is 23 K when MCLK is 6 MHz, and it can be decided by setting register ICCR.

The shift clock frequency can be decided by following formula:

FSCLK = MCLK / (m*n+2)

The parameter m and n are decided by register ICCR0, which is used in initial function. User can ignore it when using 6M MCLK and 23 KHz shift clock frequency.

3.2 Write_I²C Function

Table 3 describes Write_I²C function.

Table 3. Write_I²C Function

Function prototype	void Write_I²C (unsigned char addr, unsigned char subadd, unsigned char data)
Behavior description	Writes data to slave chip
Input parameter1	addr, address of slave chip, range is 0x00 to 0xff
Input parameter2	subadd, sub address of slave chip, range is 0x00 to 0xff
Input parameter3	data, data which write to slave chip, range is 0x00 to 0xff
Return value	None
Example	In case slave chip is EEPROM and it's address is 0xa0, then write 0x25 to EEPROM sub-address 0x01: Write_I2C(0xa0,0x01,0x25);

3.3 Read_I²C Function

Table 4 describes Read_I²C function.

Table 4. Read_I²C Function

Function name	AD_Read
Function prototype	unsigned char Read_I²C (unsigned char addr, unsigned char subaddr)
Behavior description	Read data from slave chip
Input parameter1	addr, address of slave chip, rang is 0x00 to 0xff
Input parameter2	subadd, sub address of slave chip, rang is 0x00 to 0xff
Return value	Data saved in slave chip sub-address
Example	Read data from EEPROM and sub-address is 0x01: [variable]= Read_I²C(0xa0,0x01);

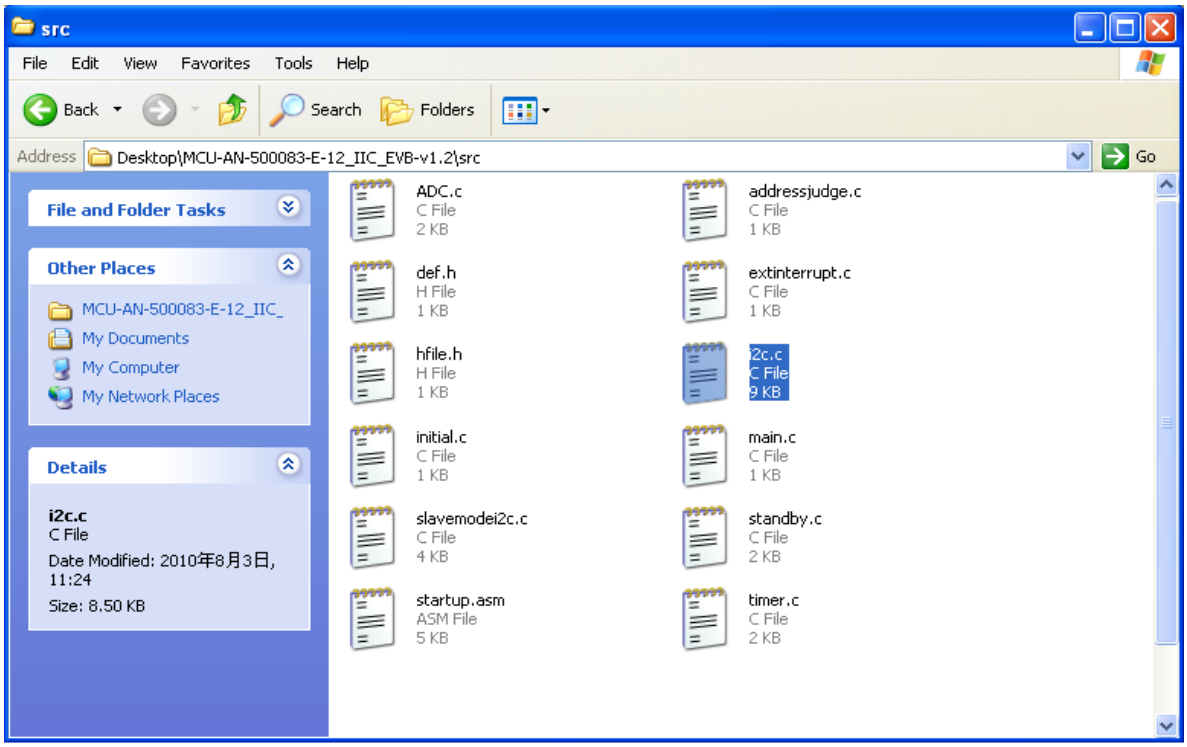
4 Usage Demo

This section describes the steps of how to use this library and some usage cautions.

4.1 Usage Steps of MB95F310 MCU

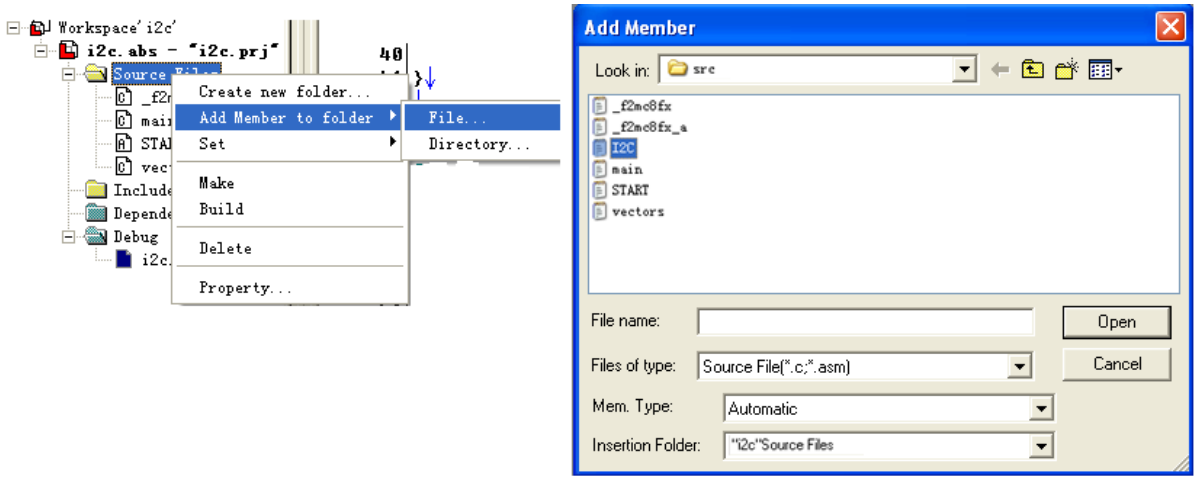
- First step is adding I²C library to project document, Figure 1 is a sample for adding this library.

Figure 1. First Step of Use I2C Library



- Second step is adding I²C library to project, Figure 2 is a sample for this work

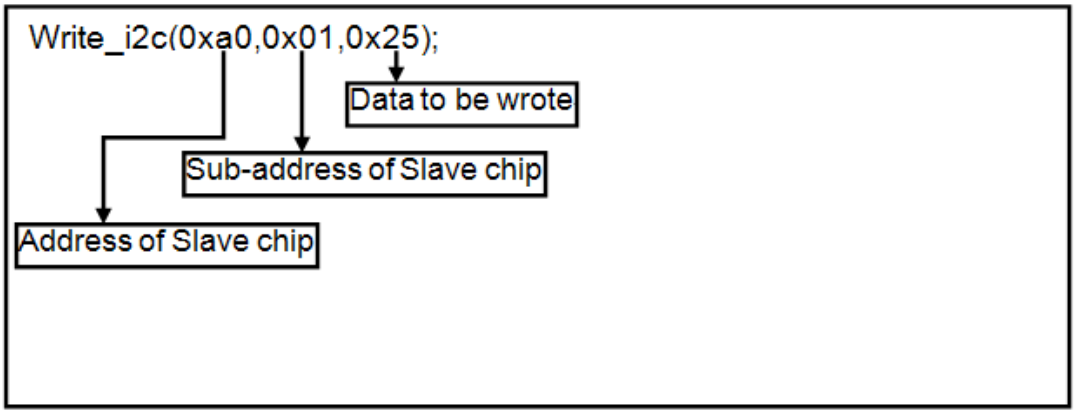
Figure 2. Second Step of Use I2C Library



- Third step is using the I²C library, for detailed library please refer to Table 1. About the initial function it is used in the library internally, so user can ignore it.

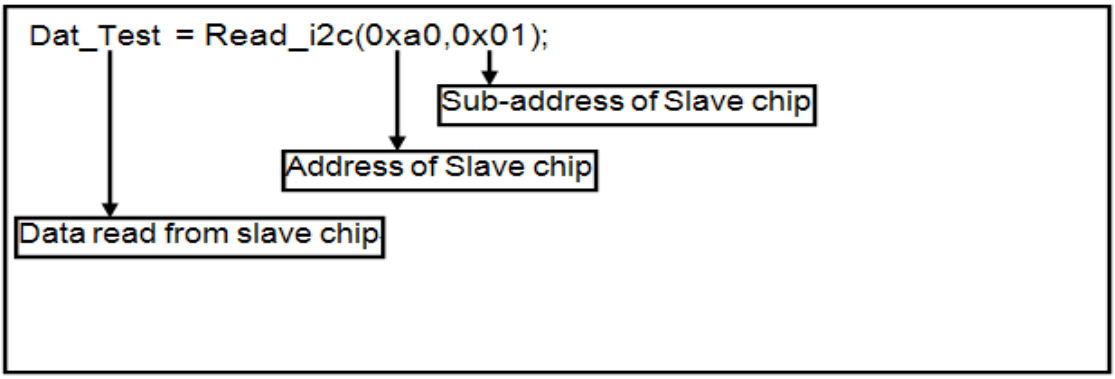
  If user wants to write data, please use Write_I²C function, Figure 3 is an example which writes data 0x25 to EEPROM sub-address 0x01.

Figure 3. Example of Writing



If user wants to read data from slave device, Figure 4 is an example which reads data from EEPROM sub-address 0x01.

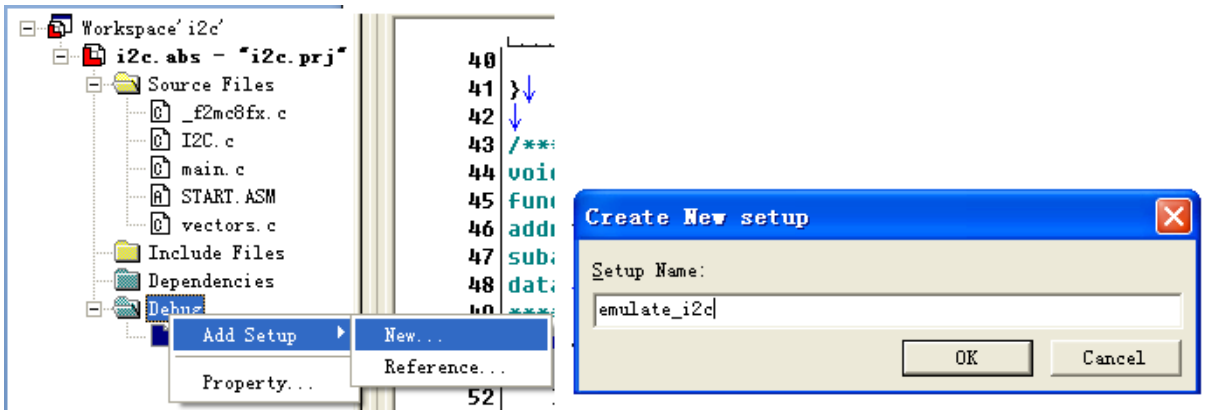
Figure 4. Example of Reading



- Fourth step is debugging,

  User needs to add debug document. Debugging mode includes serial interface, USB interface and LAN interface. Figure 5 is an example for creating serial interface debug document.

Figure 5. Step of Setting Debug Environment



For detailed debug information please refer to 5. Debugging.

4.2 Software Usage Attention

When writing data to slave device, if the slave device didn't generate acknowledge sign to master, the following items may cause it:

- The connect of I^2C is not steady
- The power of slave chip maybe not normal
- I^2C is not initialized, I^2C bus is busy
- Shift frequency clock is not suitable for slave chip

4.3 Sequence of Data Transfer

When transmitting, the MSB bit is transmitted first.

5 Debugging

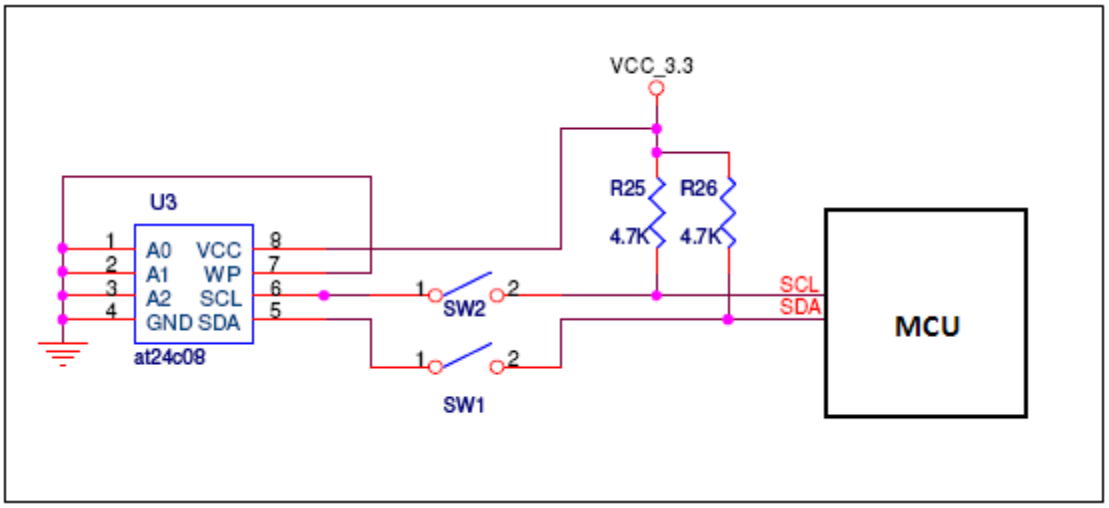
This section describes how to debug the sample code I²C.prj on start kit and what will happen when the code is running.

Attachment is the sample project I²C.prj.

This project is based on our start-kit MB2146-450-E and the target MCU is MB95F310. For detailed information, you can access our website or contact with our sales window person.

When debugging, user must connect slave device and master chip by I²C. Figure 6 describes the connecting.

Figure 6. Hardware Description

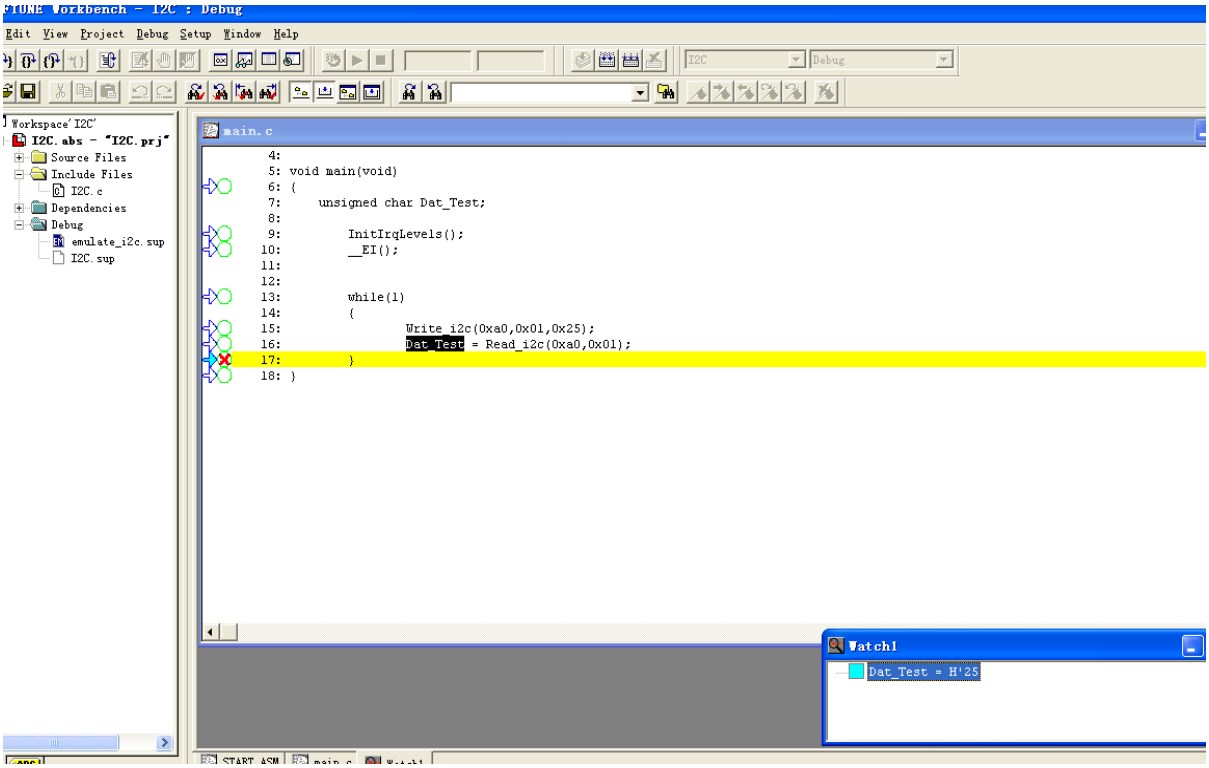


The value of pull up resister R25 and R26 can select from 1 KΩ to 10 KΩ.

When debugging user can write or read data from EEPROM. In this project global variable Dat_Test is used to save the data read from slave device.

Figure 7 is a debugging picture. User can see read result in watch window.

Figure 7. See Variable Dat_Test



6 Additional Information

For more information about how to use MB95310 EV-board, BGM Adaptor and SOFTUNE, please refer to EV-Board MB2146-450-E User Manual, or visit website:

http://www.cypress.com/documentation/application-notes/mb95310370-mb2146-450-e-lcd-evb-user-manual

Document History

Document Title: AN204730 – F²MC-8FX Family 8-Bit Microcontroller MB95310/370 Series I²C Library Usage API

Document Number: 002-04730

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	-	HUAL	11/10/2009	Initial release
			03/17/2011	Modified Chinese remark
*A	5249905	HUAL	05/18/2016	Migrated Spansion Application Note MCU-AN- 500052-E-10 to Cypress format.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Lighting & Power Control	cypress.com/powerpsoc
Memory	cypress.com/memory
PSoC	cypress.com/psoc
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless/RF	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corporation. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation, 2009-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.
TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.
Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.