CE224406 – PSoC 4 UART

Objective

This example shows how to use the PSoC® Creator™ Serial Communication Block (SCB) Component configured as a UART in a PSoC 4 device.

Requirements

Tool: PSoC Creator 4.2

Programming Language: C (Arm® GCC 5.4.1)

Associated Parts: PSoC 4 family. DMA example only works with DMA enabled controllers

Related Hardware:

- Polling and Interrupt projects: CY8CKIT-042 PSoC 4 Pioneer Kit
- DMA project: CY8CKIT-044 PSoC 4 M-Series Pioneer Kit

Overview

This example contains three PSoC Creator projects that utilize a UART Component. The projects demonstrate polling, interrupt, and DMA methods for UART data communication. Each project implements full-duplex communication with a PC using the USB-UART bridge in a Cypress PSoC 4 development kit.

Hardware Setup

This code example is set up for CY8CKIT-042 or CY8CKIT-044. If you are using a different kit, see Reusing This Example.

For CY8CKIT-042, the USB-UART bridge in KitProg2 module is used:

1. Connect the \UART:rx\ pin P0[4] to P12[7] on header J8.
2. Connect the \UART:tx\ pin P0[5] to P12[6] on header J8.

Other kits use different pins for the UART. Make sure that you select the pins that are right for your kit.

There are no connections to make for CY8CKIT-044 for the DMA example.

Software Setup

This design requires a terminal emulator such as PuTTY or Tera Term running on your computer.

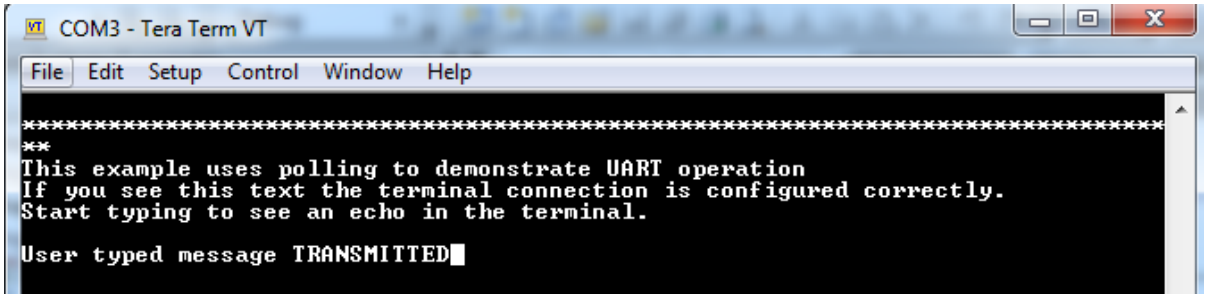
Operation

Follow these steps to communicate with the PC host:

1. Make sure you connect the correct pins for your kit, as noted in the Hardware Setup section.
2. Connect CY8CKIT-042 or CY8CKIT-044 to your computer using a USB cable.
3. Build the project and program it into the PSoC 4 device. Choose **Debug** > **Program**. For more information on device programming, see PSoC Creator Help.
4. Open a terminal emulator on your computer and configure the program to the appropriate COM port. Configure the baud rate to 115200, data bits to 8, no parity bits, stop bit as 1, and no control flow.

5. Press the reset button on the kit and observe the welcome message printed on the terminal program, as shown in Figure 1.

Figure 1. Message Printed on the Terminal



6. Type a character and observe the terminal program echoing the typed character.

Design and Implementation

There are three projects in this example:

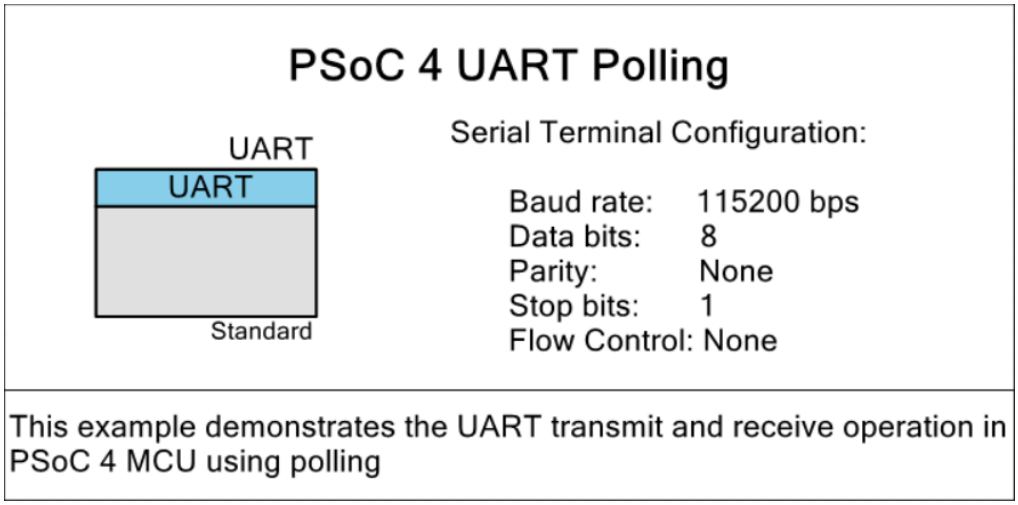
UART_ Polling

In the UART _Polling example, the following functions are performed:

1. The SCB Component is configured as a UART.
2. The UART transmits a welcome message through a terminal.
3. Checks if a new character is received (sent from keyboard inputs) from terminal.
4. If a character is received, the received character is echoed to the terminal.

The top-level design of the PSoC Creator project is shown in Figure 2.

Figure 2. UART_Polling Top Design Schematic



There are two methods to implement polling; the first method uses non-blocking function calls and the second method uses blocking function calls.

UART_ Interrupt

In the UART_Interrupt example, the following functions are performed:

1. The SCB Component is configured as a UART.
2. UART interrupts are configured and enabled with the `ISR_UART` function as the interrupt handler.
3. UART transmits a welcome message through a terminal.
4. UART continuously checks whether the flags for the received character or UART error are HIGH. If HIGH, the character is transmitted via UART or the error is handled (LED turns ON).
5. Once an interrupt event occurs (Interrupting on “UART RX FIFO not empty”), the Interrupt Service Routine (ISR) is called.

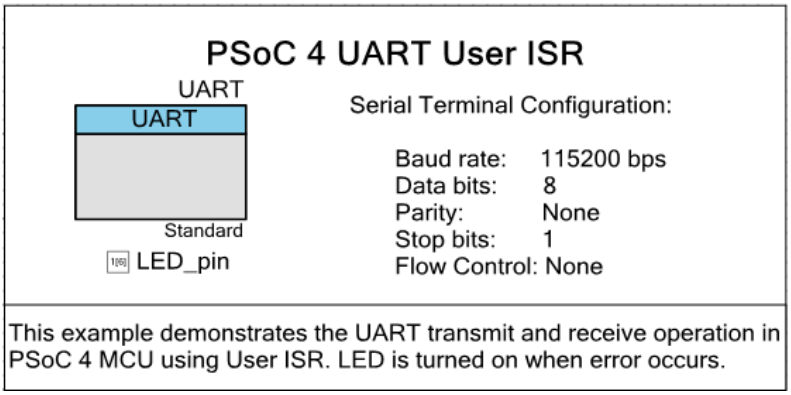
The ISR_UART function does the following:

1. Confirms the identity of the interrupt source and event.
2. Sets the `received_character` flag and saves the character; or sets the error flag.

It is important to save the character before clearing the interrupt source.

The top-level design of the PSoC Creator project is shown in Figure 3.

Figure 3. UART_Interrupt Top Design Schematic



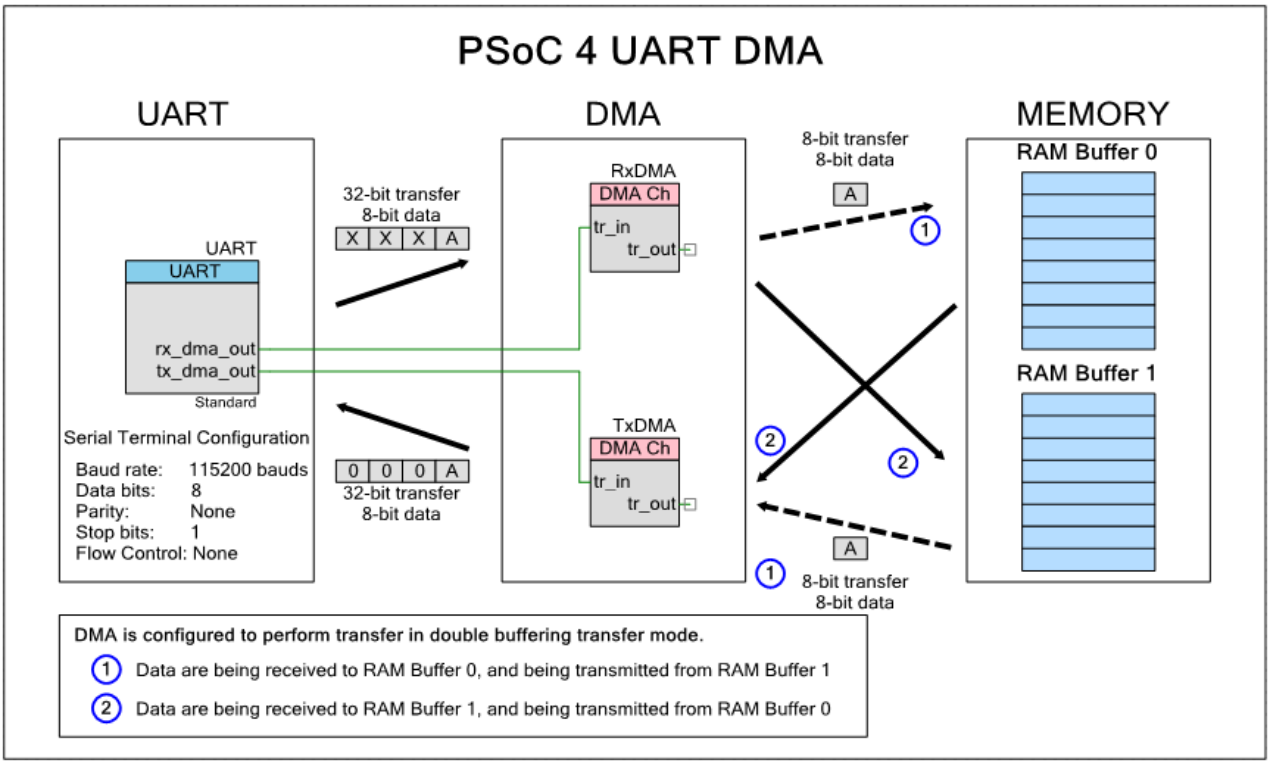
UART_DMA

In the UART_DMA example, the code performs the following tasks:

1. Configures the SCB Component as a UART.
2. Sets up the UART data buffers.
3. The UART transmits a welcome message through a terminal.
4. Sets up the receive and transmit DMA channels.
5. Checks the RxDMA interrupt source to ensure that the DMA transfer is complete.
6. Enables the transfer process and swaps the pointers of buffers/functions.

The top-level design of the PSoC Creator project is shown in Figure 4.

Figure 4. UART_DMA Top Design Schematic



In this example, the SCB Component is configured as a UART. Characters are received from the terminal and buffered through DMA channels. After buffering, the character is transmitted (echoed) back to the terminal.

Components and Settings

Table 1 lists the PSoC Creator Components used in used in this example, how they are used in the design, and the non-default settings required so they function as intended.

Table 1. PSoC Creator Components

Component	Instance Name	Purpose	Non-default Settings
UART (SCB Mode) (Polling)	UART	Handle UART serial communication	None
UART (SCB Mode) (Interrupt)	UART	Handle UART serial communication	Default Settings with different UART Advanced Settings. See Figure 5.
UART (SCB Mode) (DMA)	UART	Handle UART serial communication	Default Settings with different UART Advanced Settings. See Figure 6.
DMA Channel	RxDMA	Handle the Rx DMA channels and descriptors	See Figure 7
DMA Channel	TxDMA	Handle the Tx DMA channels and descriptors	See Figure 8

Figure 5.UART Parameter Settings (Interrupt)

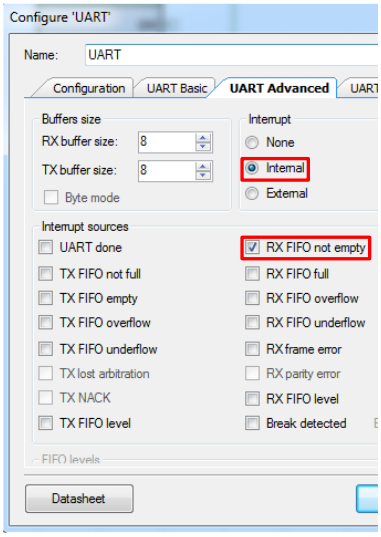


Figure 6. UART Parameter Settings (DMA)

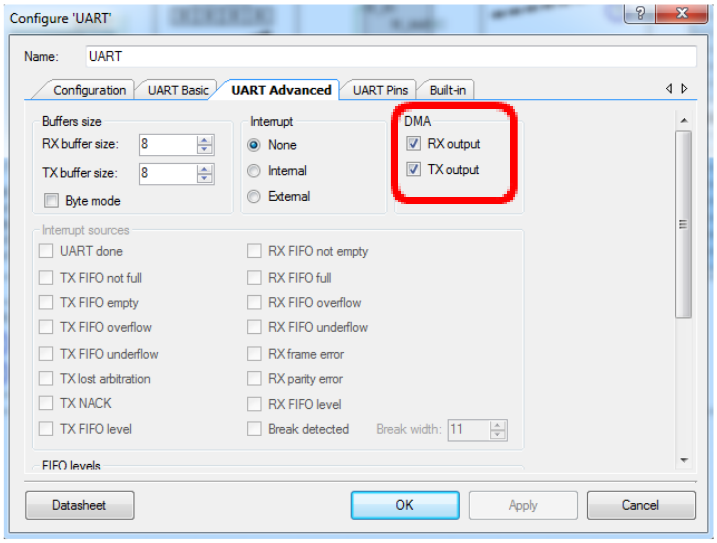


Figure 7. RxDMA Parameter Settings

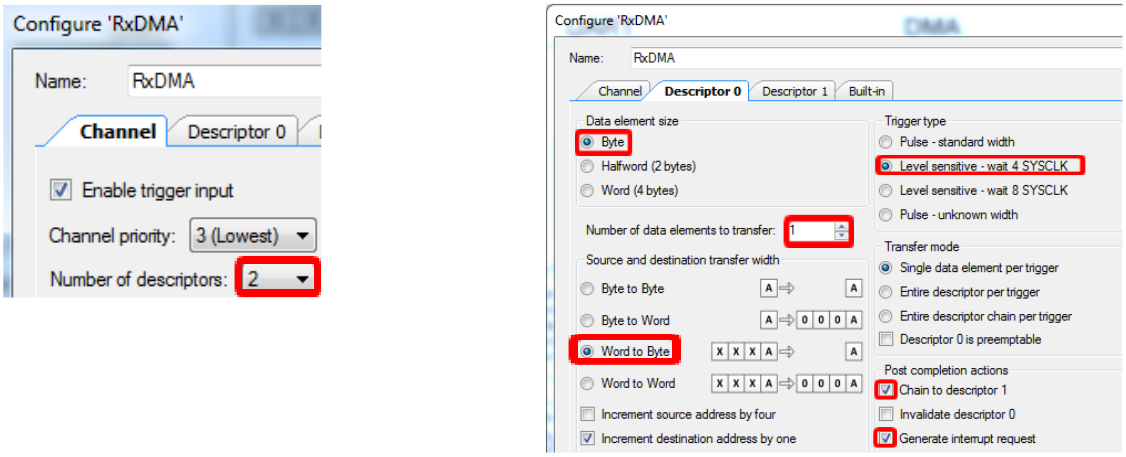
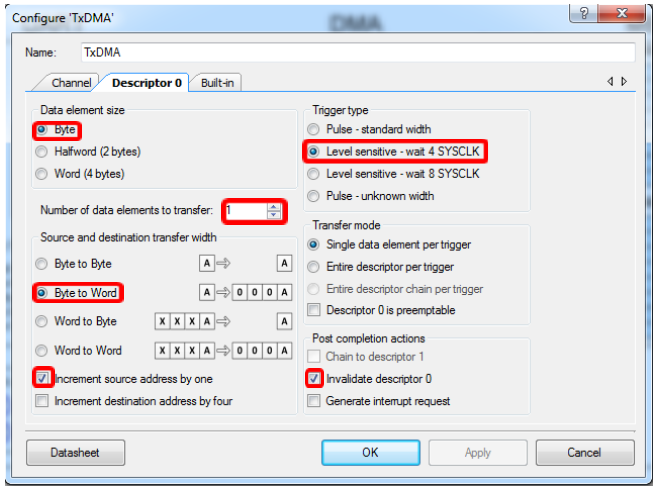


Figure 8 TxDMA Parameter Settings



For information on the hardware resources used by a Component, see the Component datasheet.

Reusing This Example

This example is designed for the CY8CKIT-042 pioneer kit. To port this design to a different PSoC 4 device, kit, or both, do the following:

1. In PSoC Creator, select **Project** > **Device Selector** to change the target device. Select your device as listed in Table 2.
2. Make sure that the **SysClk Desired frequency** is set to 24 MHz after the device is changed.
3. In the PSoC Creator Workspace Explorer, select the **Clocks** interface listed under **Design Wide Resources**.
4. Set the **SysClk Desired Frequency** to 24 MHz, if it is not already.
5. Update the pin assignments in the Design Wide Resources Pins settings as needed (see Table 3).

Table 2. Development Kits and Associated Devices

Development Kit	Device
CY8CKIT-041	CY8C4146AZI-S433
CY8CKIT-042	CY8C4245AXI-483
CY8CKIT-042-BLE	CY8C4247LQI-BL483
CY8CKIT-044	CY8C4247AZI-M485
CY8CKIT-046	CY8C4248BZI-L489
CY8CKIT-048	CY8C4A45AZI-483

Note: The DMA project works only with PSoC 4 devices with DMA. To see a list of available devices with DMA, in PSoC Creator, select **File** > **New** > **Project** > **Target Device** > **PSoC 4**, and then choose **Launch device selector** in the second drop-down menu. Find **DMA channels** in the filter and deselect **none**. This will give a list of all PSoC 4 devices with DMA.

Table 3. Pin Assignments for Different Kits

Pin Name	Development Kit					
	CY8CKIT-041	CY8CKIT-042	CY8CKIT-042-BLE	CY8CKIT-044	CY8CKIT-046	CY8CKIT-048
\UART:rx\	P0[4]	P0[4]	P1[4]	P7[0]	P3[0]	P0[4]
\UART:tx\	P0[5]	P0[5]	P1[5]	P7[1]	P3[1]	P0[5]

For the CY8CKIT-048, connect \UART:tx\ to P12[6] and \UART:rx\ to P12[7] on header J16.

In some cases, a resource used by a code example (for example, a Universal Digital Block) is not supported on another device. In that case, the example will not work. If you build the code targeted at such a device, you will get errors. See the device datasheet for information on what a device supports.

Related Documents

Application Notes	
AN79953 – Getting Started with PSoC® 4	Describes PSoC 4 and shows how to build the attached code example
Code Examples	
CE224431 – PSoC 4 UART printf	This code example shows how to re-direct and use the printf() function for sending data, using a PSoC Creator Serial Communication Block (SCB) Component configured as a UART in a PSoC 4 device.
PSoC Creator Component Datasheets	
SCB	A multifunction hardware block that implements the following communication components: I2C, SPI, UART, and EZI2C
DMA	Transfers data to and from memory, components, and registers that is independent of the CPU.
Device Documentation	
PSoC 4 Datasheets	PSoC 4 Technical Reference Manuals
Development Kit (DVK) Documentation	
CY8CKIT-042 PSoC® 4 Pioneer Kit	
CY8CKIT-044 PSoC® 4 M-Series Pioneer Kit	
PSoC 4 Kits	
Tool Documentation	
PSoC Creator	Go to the **Downloads** tab for Quick Start and User Guides

Document History

Document Title: CE224406 - PSoC 4 UART

Document Number: 002-24406

Revision	ECN	Submission Date	Description of Change
**	6260169	07/24/2018	New code example
*A	6672896	09/16/2019	Updated code example template and minor updates for clarity

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
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

Cypress Developer Community

Community | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

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

© Cypress Semiconductor Corporation, 2018-2019. This document is the property of Cypress Semiconductor Corporation and its subsidiaries ("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. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress shall have no liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. CYPRESS DOES NOT REPRESENT, WARRANT, OR GUARANTEE THAT CYPRESS PRODUCTS, OR SYSTEMS CREATED USING CYPRESS PRODUCTS, WILL BE FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (collectively, "Security Breach"). Cypress disclaims any liability relating to any Security Breach, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any Security Breach. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. 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. "High-Risk Device" means any device or system whose failure could cause personal injury, death, or property damage. Examples of High-Risk Devices are weapons, nuclear installations, surgical implants, and other medical devices. "Critical Component" means any component of a High-Risk Device whose failure to perform can be reasonably expected to cause, directly or indirectly, the failure of the High-Risk Device, 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 any use of a Cypress product as a Critical Component in a High-Risk Device. You shall indemnify and hold Cypress, its directors, officers, employees, agents, affiliates, distributors, and assigns harmless from and against all claims, costs, damages, and expenses, arising out of any claim, including claims for product liability, personal injury or death, or property damage arising from any use of a Cypress product as a Critical Component in a High-Risk Device. Cypress products are not intended or authorized for use as a Critical Component in any High-Risk Device except to the limited extent that (i) Cypress's published data sheet for the product explicitly states Cypress has qualified the product for use in a specific High-Risk Device, or (ii) Cypress has given you advance written authorization to use the product as a Critical Component in the specific High-Risk Device and you have signed a separate indemnification agreement.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, 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.