CE210988 - CY8CKIT-042-BLE F-RAM™ Data Logger

Objective

This example project is based on a PSoC® Creator™ starter design for the PSoC 4 device. It demonstrates how F-RAM can be used with the PSoC to capture and log the analog routing capability of PSoC 4 in the real time. In this project, every input has corresponding dedicated feedback networks and hence different gains. You can change the active channel by pressing a switch. In addition, the data is recorded to the on-board F-RAM device that implements a rolling buffer.

Features

- Opamp as noninverting amplifier
- ADC used in single-ended mode
- HyperTerminal displays the ADC results sent via UART
- LED indicates when the ADC input is outside the defined voltage window
- Analog Muxes multiplex three inputs and their corresponding gains
- Debouncer detects valid switch presses
- F-RAM Data Logger

Development Kit Configuration

This example project is designed to run on the CY8CKIT-042 kit from Cypress Semiconductor. A description of the kit, along with more example programs and ordering information, can be found at http://www.cypress.com/go/cy8ckit-042.

The project requires changes to configuration settings to run on other kits from Cypress Semiconductor. Table 1 lists the supported kits. To switch from CY8CKIT-042 to any other kit, change the project's device with the help of Device Selector called from the project's context menu.

Table 1. Development Kits Versus PSoC Parts

Development Kit	Device
CY8CKIT-042	CY8C4245AXI-483
CY8CKIT-042-BLE	CY8C4247LQI-BL483
CY8CKIT-044	CY8C4247AZI-M485
CY8CKIT-046	CY8C4248BZI-L489

Table 2 lists the pin assignments for supported kits.

Table 2. Pin Assignments for Supported Kits

Pin Name	Development Kit			
	CY8CKIT-042	CY8CKIT-042 BLE	CY8CKIT-044	CY8CKIT-046
A_Out	P1[2]	P1[2]	P1[2]	P1[2]
UART:TX	P0[5]*	P1[5]	P7[1]	P3[1]
LED	P1[6]	P2[6]	P0[6]	P5[2]
Input_1	P2[0]	P2[0]	P2[0]	P2[0]
Input_2	P2[1]	P2[1]	P2[1]	P2[1]
Input_3	P2[2]	P2[2]	P2[2]	P2[2]
G1	P2[3]	P2[3]	P2[3]	P2[3]
G2	P2[4]	P2[4]	P2[4]	P2[4]
G3	P2[5]	P2[5]	P2[5]	P2[5]
In_Sel	P0[7]	P2[7]	P0[7]	P0[7]

* Connect P0[5] to pin P12[6] on header J8.

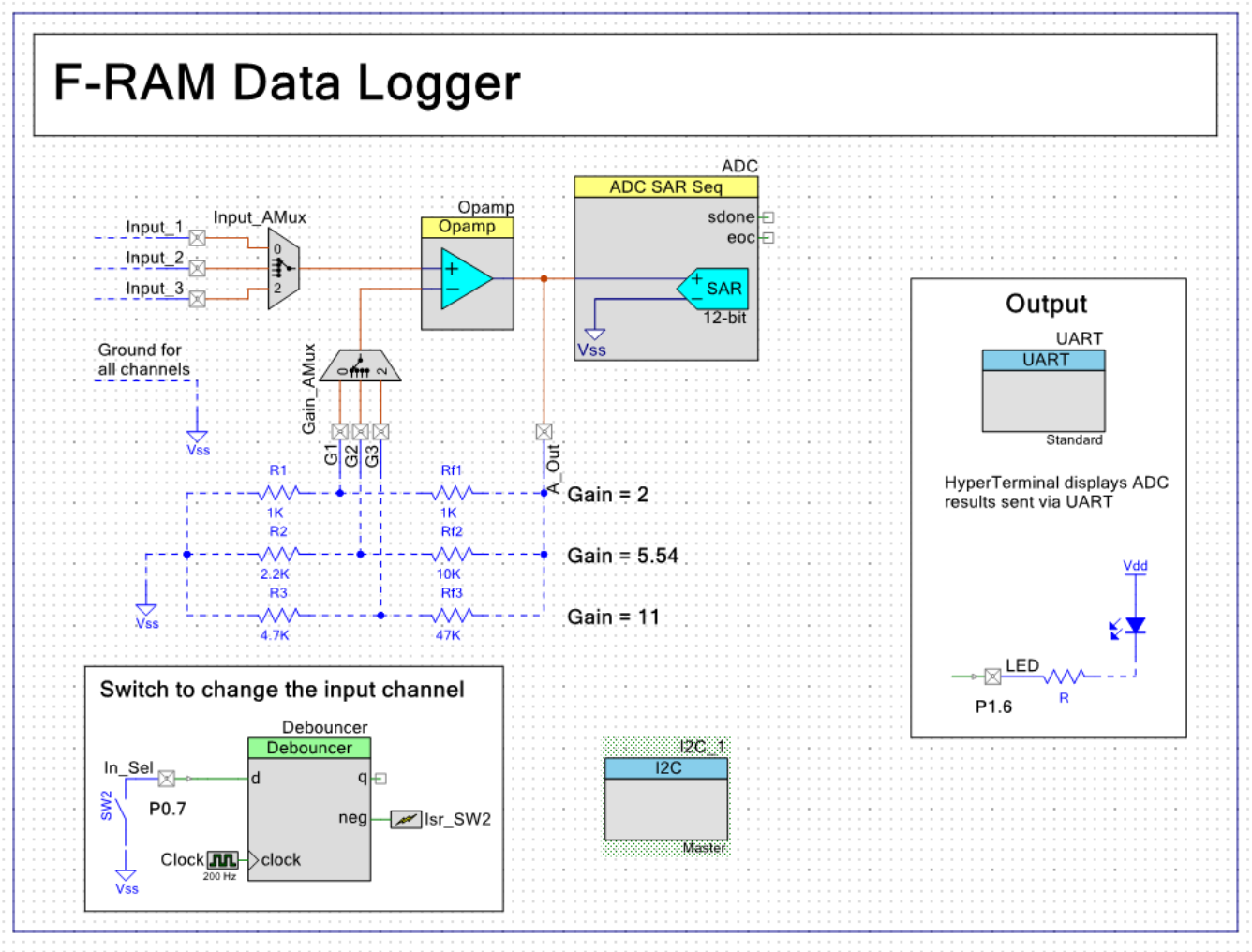
The following configuration instructions provide a guideline to test this design. For simplicity, the instructions describe the step-wise process to be followed when testing this design with the PSoC 4 Pioneer Kit (CY8CKIT-042).

1. Set jumper J9 (J16 for CY8CKIT-042-BLE) to the 5.0V position.
2. Connect three input signals to Input_1, Input_2, and Input_3.
3. Connect all the external resistors as shown in the top design schematic.
4. Connect a USB cable to the PSoC 4 Pioneer Kit DVK and PC with the HyperTerminal program.

Project Configuration

This example project consists of ADC SAR Seq, Opamp, AMuxSeq, UART, and Debouncer Components. The top design schematic is shown in Figure 1. The opamp is used to amplify the input signal; the input channel is selected using Input_AMux and the feedback network is selected using Gain_AMux. A UART is used to send ADC results to HyperTerminal. Debouncer is used to remove glitches from the input switch. The SAR ADC converts the analog output of the Opamp, into digital values. The ADC also generates an interrupt when its input is outside the defined voltage window (250 mV - 750 mV). The LED turns ON when the ADC generates this interrupt. The UART Component sends the ADC output of the active channel along with the channel number to HyperTerminal.

Figure 1. Top Design Schematic



The opamp is configured in high stability, high power, and 10-mA output current mode. The ADC is configured in single-ended mode. The ADC averages 256 consecutive samples to produce the final result. The ADC Component configuration is shown in Figure 2. Switch SW2 (P2.7) and pin In_Sel are used to change the analog mux channels; Input_AMux selects the input channel and Gain_AMux selects the corresponding gain. The I2C Master is used to move data to and from the F-RAM device. The I2C Component configuration is shown in Figure 3.

Figure 2. ADC Configuration Window

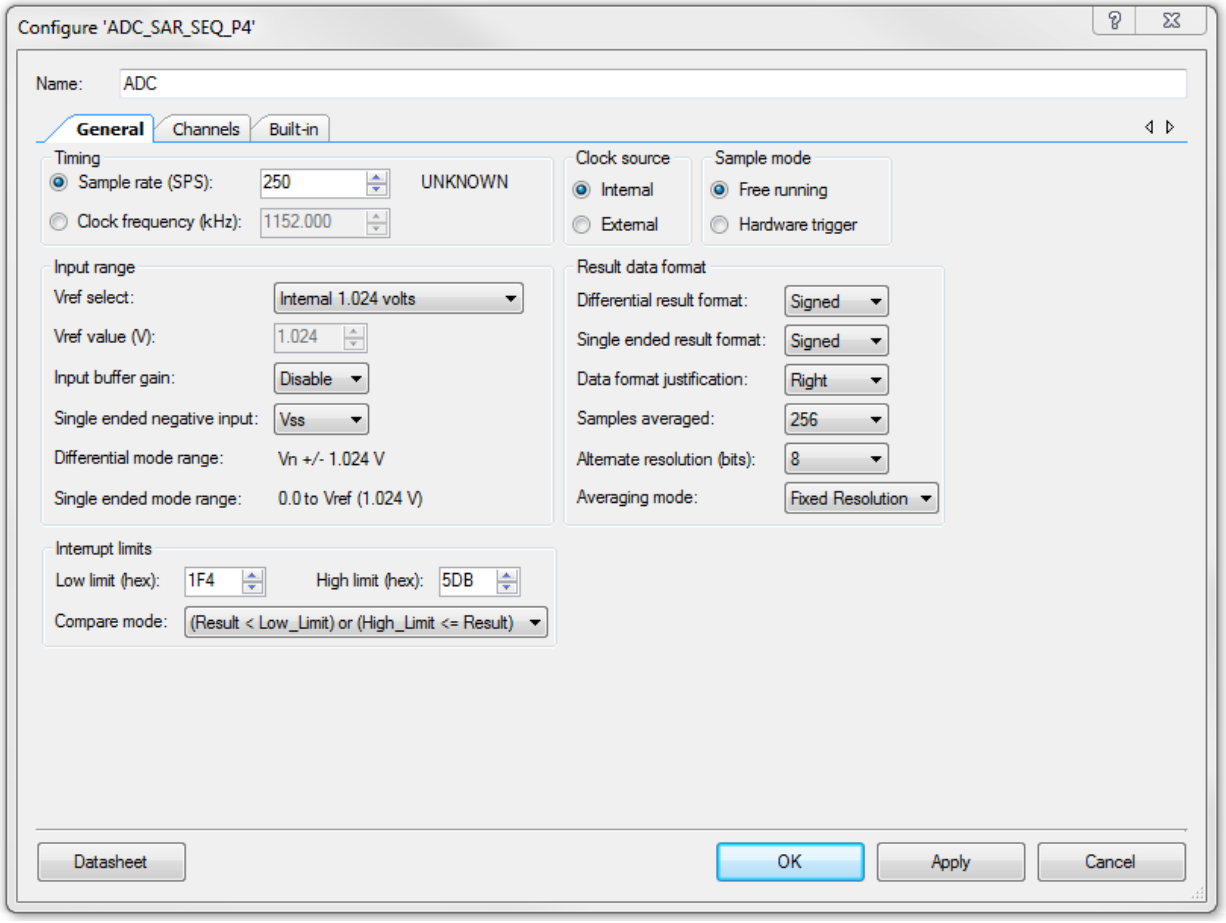
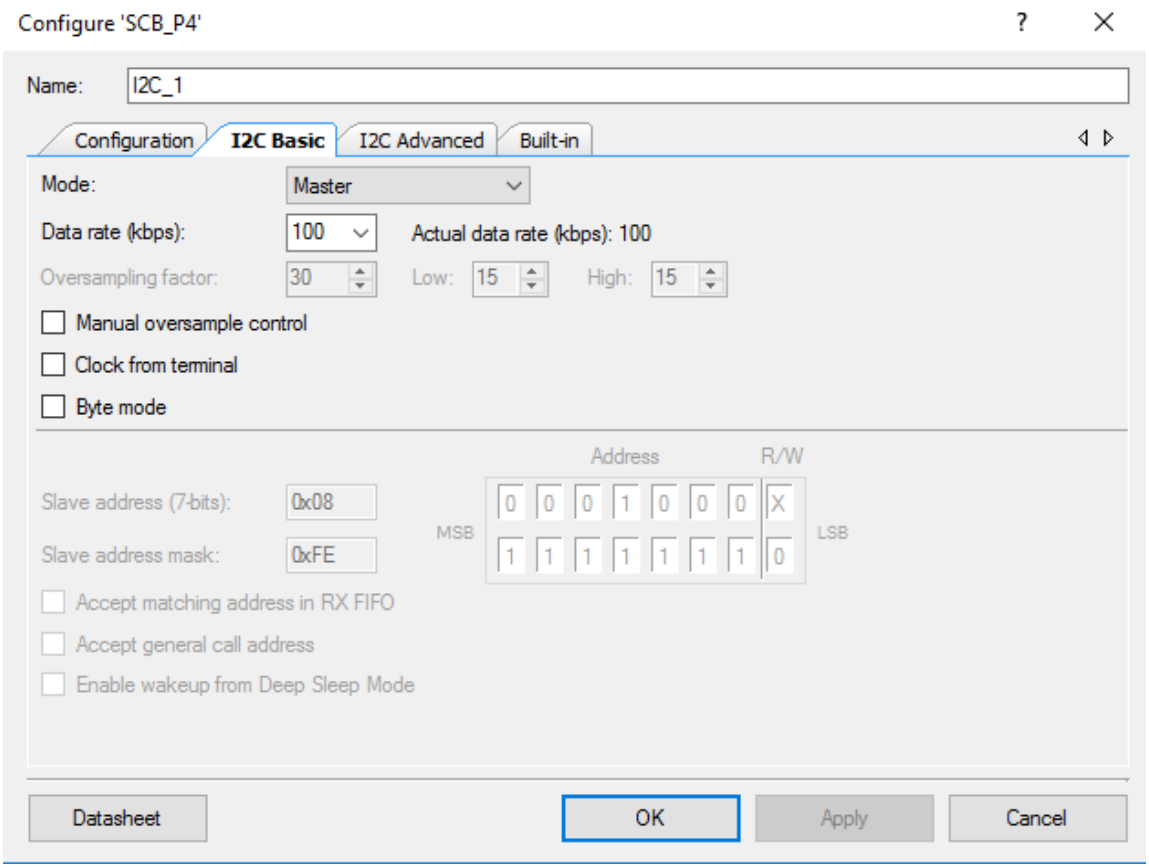


Figure 3. I2C Configuration Window



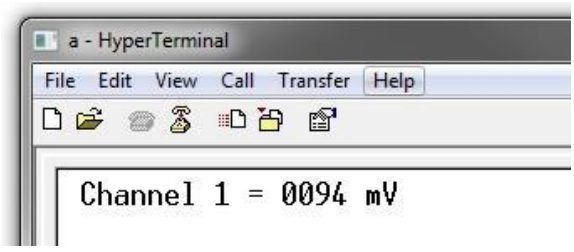
Project Description

In the main function, all Components are started, both the analog muxes are initialized to select the channel zero, and ADC conversion is started. The “for” loop in *main.c* waits for the ADC to finish the conversion. When the ADC result is available, it is sent through UART to HyperTerminal. The ADC continuously generates interrupts when its input is outside the defined voltage window (250 mV - 750 mV). This interrupt is used to control an LED. This LED is turned ON when the ADC input is outside the window.

Expected Results

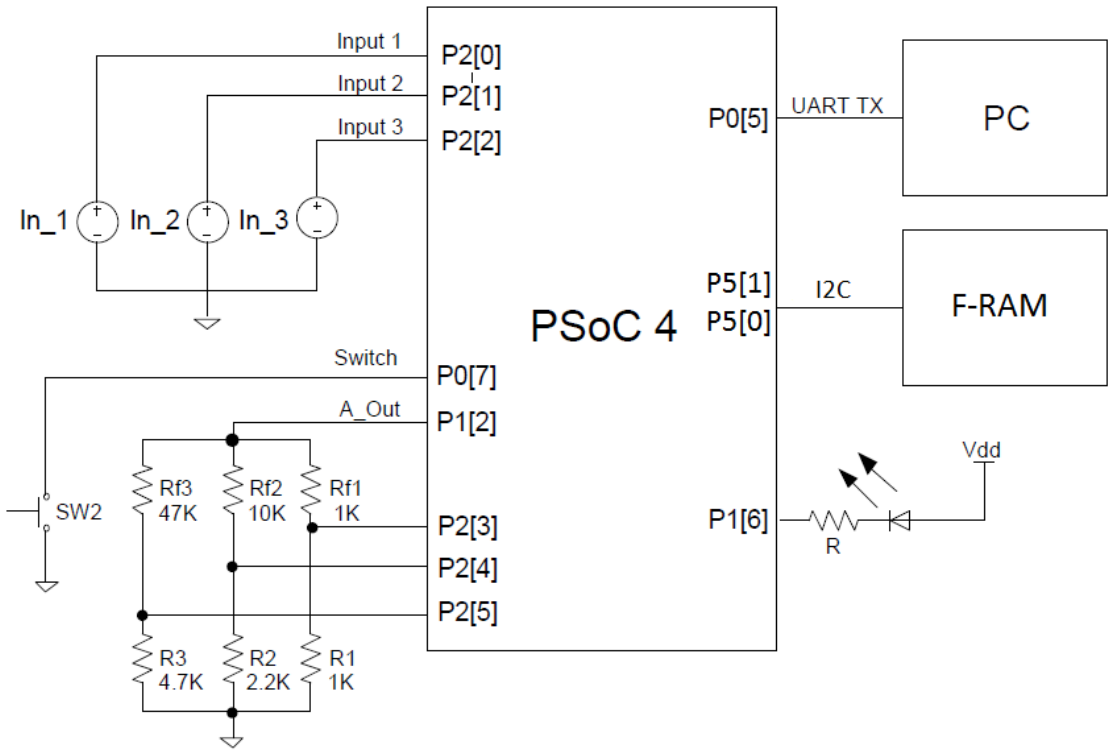
The analog input channel should change when switch SW2 (P2.7) is pressed. HyperTerminal displays the active channel and its input voltage. The LED will turn on when the ADC result is outside the voltage window (250 mV – 750 mV).

Figure 4. Result



Schematic

Figure 5. Connection Schematic



Note: If the LED is active HIGH, replace LOW with HIGH to turn the LED ON and vice-versa. The section of the code that needs to be changed in *main.c* is mentioned below:

```
/* Turn ON the LED when the input is outside the window (250mV - 750mV) */
LED_Write(LOW);

/* Turn OFF the LED when the input is within the defined window (250mV - 750mV) */
LED_Write(HIGH)
```

Using UART to Communicate with a PC Host

This example project communicates with a PC host using UART. The HyperTerminal program is required in the PC to communicate with PSoC 4. If you do not have the HyperTerminal program installed, download and install any serial port communication program. Free programs such as HyperTerminal and Bray's Terminal are available on the Web.

Follow these steps to communicate with the PC host.

1. Connect the USB cable between the PC and PSoC 4 Pioneer Kit.
2. Open Device Manager in your PC, find the COM port in which the PSoC 4 is connected, and note the port number.
3. Open the HyperTerminal program and select the COM port in which PSoC 4 is connected.
4. Configure the baud rate, parity, stop bits, and flow control information in the HyperTerminal configuration window. These settings should match the configuration of the PSoC Creator UART Component in the project.
5. Start communicating with the device as explained in the project description.

Using I2C to Communicate with the F-RAM Device

This example project communicates with the F-RAM device using I2C. I2C is a simple Clock/Data Bus using 8-bit serial words. Code 1 and Code 2 show the Write and Read operations for the I2C master to access the F-RAM device.

Code 1. I2C F-RAM Write

```
/*******************************************************************************

* Function Name: FRAM_Write
********************************************************************************
*
* Summary:
*  F-RAM Byte Write Command
* uint16 address - address in the F-RAM array
* uint8 data - data to be written
*
*******************************************************************************/
void FRAM_Write(uint16 address, uint8 data)
{
    uint8 txBuffer[BUFFER_SIZE];

    txBuffer[0] = address >> 8;
    txBuffer[1] = address;
    txBuffer[2] = data;


    /* Clear any previous status */
    I2C_1_I2CMasterClearStatus();

    /* I2C Wrtie Command */
    status = I2C_1_I2CMasterWriteBuf(SLAVE_ADDRESS, (uint8 *) txBuffer, BUFFER_SIZE,
I2C_1_I2C_MODE_COMPLETE_XFER);

    for(;;)
    {
        if(0u != (I2C_1_I2CMasterStatus() & I2C_1_I2C_MSTAT_WR_CMPLT))
        {
            /* Transfer complete. Check Master status to make sure that transfer
            completed without errors. */
            break;
        }
    }
}
```

Code 2. I2C F-RAM Read

```
/*******************************************************************************
* Function Name: FRAM_Read
********************************************************************************
*
* Summary:
*  F-RAM Byte Read Command
* uint16 address - address in the F-RAM array
* uint8 retrun - data to be read
*
*******************************************************************************/
uint8 FRAM_Read(uint16 address)
{
    uint8 rxBuffer[BUFFER_SIZE-2];
    uint8 txBuffer[2];

    txBuffer[0] = address >> 8;
    txBuffer[1] = address;

    /* Clear any previous status */
```

```
    I2C_1_I2CMasterClearStatus();

    /* I2C Wrtie Command to Set the Address*/
    status = I2C_1_I2CMasterWriteBuf(SLAVE_ADDRESS, (uint8 *) txBuffer, 0x2,
I2C_1_I2C_MODE_COMPLETE_XFER);
    for(;;)
    {
        if(0u != (I2C_1_I2CMasterStatus() & I2C_1_I2C_MSTAT_WR_CMPLT))
        {
            /* Transfer complete. Check Master status to make sure that transfer
            completed without errors. */
            break;
        }
    }

    /* Clear any previous status */
    I2C_1_I2CMasterClearStatus();


    /* I2C Read Command */
    status = I2C_1_I2CMasterReadBuf(SLAVE_ADDRESS, (uint8 *) rxBuffer, 1, I2C_1_I2C_MODE_COMPLETE_XFER);
    for(;;)
    {
        if(0u != (I2C_1_I2CMasterStatus() & I2C_1_I2C_MSTAT_RD_CMPLT))
        {
            /* Transfer complete. Check Master status to make sure that transfer
            completed without errors. */
            break;
        }
    }
    return rxBuffer[0];
}
```

Document History

Document Title: CE210988 - CY8CKIT-042-BLE F-RAM™ Data Logger

Document Number: 002-10988

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5112947	JLTO	02/18/2016	New spec
*A	5734196	AESATMP9	05/11/2017	Updated logo and copyright.

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

Cypress Developer Community

Forums | WICED IOT Forums | 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, 2016-2017. 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, 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.