

## Objective

This example demonstrates on-the-fly encryption of data using Serial Memory Interface(SMIF) in MMIO and XIP modes. This example uses the CY15B104QSN Excelon™-Ultra 4-Mbit (512K × 8) Quad SPI F-RAM™ in SPI mode.

## Requirements

**Tool:** [ModusToolbox™ 1.1](#)

**Programming Language:** C

**Associated Parts:** All [PSoC® 6 MCU](#) parts

**Related Hardware:** [PSoC 6 WiFi-BT Pioneer Kit](#)

## Overview

This example demonstrates on-the-fly encryption of data using Serial Memory Interface(SMIF) in MMIO and XIP modes. This example uses the CY15B104QSN Excelon™-Ultra 4-Mbit (512K × 8) Quad SPI F-RAM in SPI mode that is available on the PSoC 6 WiFi-BT Pioneer Kit.

Plain data is encrypted in firmware and written to F-RAM in MMIO mode. The encrypted data is then read back, decrypted and compared with plain data for integrity. The firmware then shifts the SMIF operation to XIP mode. In this mode, plain data is directly written to the F-RAM. Encryption is performed on-the-fly by the SMIF block. Data is read back directly and compared with the written data for integrity check; decryption of data is performed on-the-fly by the SMIF block.

## Hardware Setup

This example uses the PSoC 6 WiFi-BT Pioneer Kit's default configuration. Refer to the kit guide to ensure that the kit is configured correctly.

**Note:** The PSoC 6 WiFi-BT Pioneer kit ships with KitProg2. ModusToolbox only works with KitProg3. Before using this code example, make sure that the kit is upgraded to KitProg3. See **ModusToolbox Help > ModusToolbox IDE Documentation > User Guide**; section PSoC 6 MCU KitProg Firmware Loader. If you do not upgrade, you will see an error like “unable to find CMSIS-DAP device” or “KitProg firmware is out of date”.

## Software Setup

This example uses [Tera Term as the](#) terminal emulator program. Install one on your PC if you don't have one.

## Operation

1. Connect the Pioneer board to your PC using the provided USB cable through the USB connector.
2. Open a terminal program and select the KitProg3 COM port. Set the serial port parameters to 8N1 and 115,200 baud.
3. Import the application into a new workspace. See [KBA225201](#).
4. Program the PSoC 6 MCU device. Select the 'mainapp' project. In the **Quick Panel**, scroll down, and click **Program (Kitprog3)**.
5. After programming, the application starts automatically. Confirm that verbose messages regarding firmware operation are displayed on the UART terminal. Figure 1 is a snapshot of the serial terminal output.

Figure 1. Serial Terminal Output

```

COM6 - Tera Term VT
File Edit Setup Control Window Help
[Info] UART initialization complete
CE227032 - PSoC 6 MCU SMIF On-The-Fly Encryption in MMIO and XIP Modes
*****
[Info] SMIF interrupt initialization complete
[Info] SMIF initialization complete
[Info] F-RAM memory slot initialization complete
[Info] SMIF operation in SPI mode
=====
[Info] Performing encryption in MMIO mode and writing data to F-RAM
[Info] Plain Data: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E
0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
[Info] Data encryption complete
[Info] Encrypted Data: 0x84 0xAC 0x28 0x5A 0xE3 0xD6 0x5A 0x05 0x41 0xF7 0x83 0xE6 0x16 0x94 0x59
0xDE 0x8B 0x5E 0x15 0xD0 0x4A 0x71 0xA4 0xF3 0x95 0xE0 0x3C 0xE2 0x81 0x0C 0x4D 0xCC
[Info] Written Data: 0x84 0xAC 0x28 0x5A 0xE3 0xD6 0x5A 0x05 0x41 0xF7 0x83 0xE6 0x16 0x94 0x59
0xDE 0x8B 0x5E 0x15 0xD0 0x4A 0x71 0xA4 0xF3 0x95 0xE0 0x3C 0xE2 0x81 0x0C 0x4D 0xCC
[Info] Received Status: 0x2
[Info] Received Data: 0x84 0xAC 0x28 0x5A 0xE3 0xD6 0x5A 0x05 0x41 0xF7 0x83 0xE6 0x16 0x94 0x59
0xDE 0x8B 0x5E 0x15 0xD0 0x4A 0x71 0xA4 0xF3 0x95 0xE0 0x3C 0xE2 0x81 0x0C 0x4D 0xCC
[Info] Data decryption complete
[Info] Decrypted Data: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E
0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
[Info] Received data matches with written data
=====
[Info] Writing zeros to F-RAM
[Info] Written Data: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
[Info] Received Status: 0x2
[Info] Received Data: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
[Info] Received data matches with written data
=====
[Info] Setting SMIF to XIP mode and writing data to F-RAM. Data is encrypted on-the-fly
[Info] Plain Data: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E
0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
[Info] Reading on-the-fly encrypted data using MMIO mode
[Info] Received Data: 0x84 0xAC 0x28 0x5A 0xE3 0xD6 0x5A 0x05 0x41 0xF7 0x83 0xE6 0x16 0x94 0x59
0xDE 0x8B 0x5E 0x15 0xD0 0x4A 0x71 0xA4 0xF3 0x95 0xE0 0x3C 0xE2 0x81 0x0C 0x4D 0xCC
[Info] Setting SMIF back to XIP mode and reading data from F-RAM. Data is decrypted on-the-fly
[Info] Decrypted Data: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E
0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F
[Info] Received data matches with written data
=====
  
```

## Debugging

You can debug the example to step through the code. Use Debug (KitProg3) configuration in the Quick Panel. See [KBA224621](#) in the Cypress community to learn how to start a debug session with ModusToolbox IDE.

## Design and Implementation

Encryption and decryption of data in the SMIF block are based on the AES-128 forward block cipher. The secret private key is programmed into the CRYPTO\_KEYx registers of the block. In the crypto block of SMIF, by applying AES-128 with 128-bit secret key to 128-bit plain text, you get 128-bit cipher text. Note that the AES-128 block cipher is on the address of the data and not on the data itself.

When using MMIO mode, plain text is encrypted by firmware. This is done by providing the address of the memory as plain text to the crypto block and then the data is XORed with the resulting cipher text. The encrypted data is then written to F-RAM. The data from F-RAM is similarly decrypted. See the code snippet below. Data is encrypted and decrypted using the Cy\_SMIF\_Encrypt API function.

```
PrintArray("[Info] Plain Data:\t", txBuffer, PACKET_SIZE);
status = Cy_SMIF_Encrypt(KIT_QSPI_HW, CY15B104QSN_SlaveSlot_2.baseAddress, encryptedtxBuffer,
PACKET_SIZE, &KIT_QSPI_context);

/* Write the sample bytes in encryptedtxBuffer to the external memory address specified */
WriteMemory(KIT_QSPI_HW, (cy_stc_smif_mem_config_t*) smifMemConfigs[0], &KIT_QSPI_context,
encryptedtxBuffer, PACKET_SIZE, extMemAddress);

/* Read data from external memory into rxBuffer and decrypt it*/
ReadMemory(KIT_QSPI_HW, (cy_stc_smif_mem_config_t*) smifMemConfigs[0], &KIT_QSPI_context, rxBuffer,
PACKET_SIZE, extMemAddress);
status = Cy_SMIF_Encrypt(KIT_QSPI_HW, CY15B104QSN_SlaveSlot_2.baseAddress, rxBuffer, PACKET_SIZE,
&KIT_QSPI_context);
```

In XIP mode, the SMIF crypto block supports on-the-fly encryption and decryption. The address accessed using XIP mode is used as plain text; the resulting cipher text is then used to write to external F-RAM on-the-fly and is not software accessible. In XIP mode, the resulting ciphertext (of the encrypted address) is XORed with the memory transfer's read data or write data.

For more information on encryption in SMIF block, see the device TRM.

The Serial Communication Block (SCB) resource is configured as a UART at 115200 baud, 8N1. It is connected to a clock operating at 923 kHz to generate the correct baud rate. The RX is on pin P5[0] and TX is on P5[1] to match the pin usage on the kit.

The resources are configured and application code runs on the CM4 CPU.

To see all the settings, review the *design.modus* file in the application.

## Resources and Settings

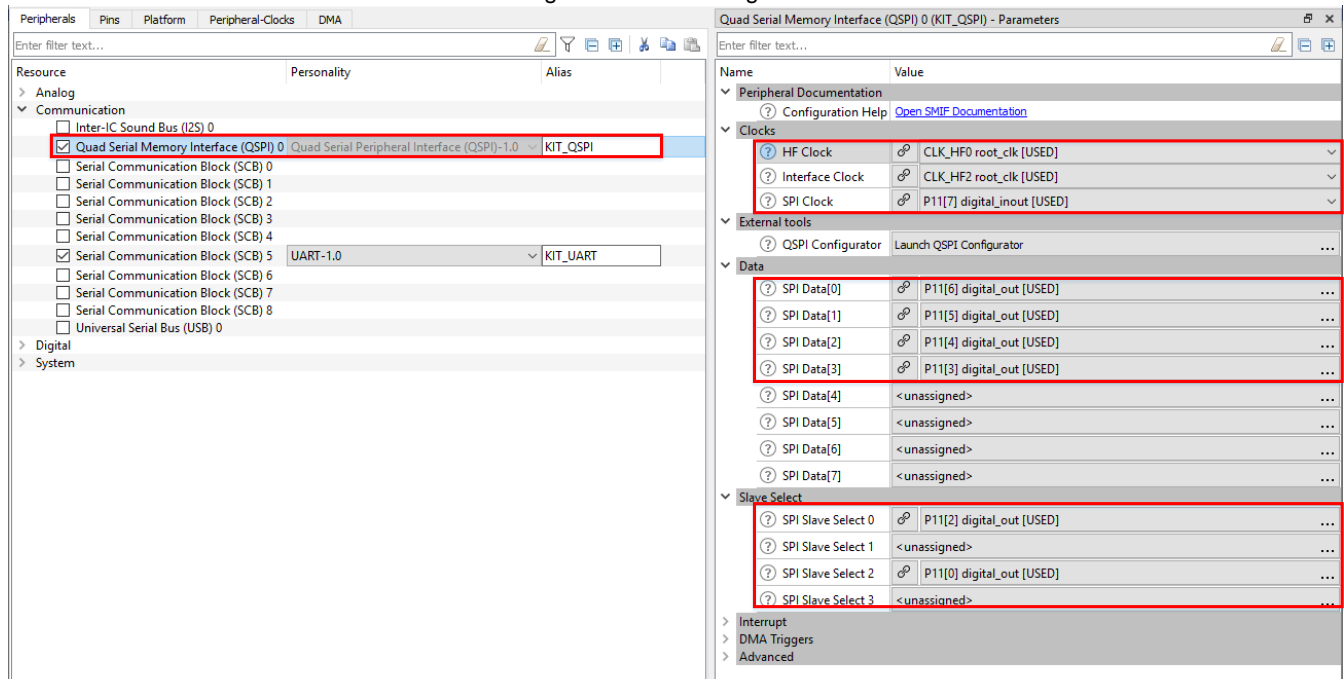
Table 1 lists the resources used in this example, and how they are used in the design.

Table 1. ModusToolbox Resources

Resource	Alias	Purpose	Non-default Settings
Quad Serial Memory Interface	KIT_QSPI	Communicates with SPI F-RAM	See <a href="#">Figure 2</a> See <a href="#">Figure 3</a> for FRAM configuration
SCB	KIT_UART	Prints messages to a terminal window	See <a href="#">Figure 4</a>
Digital Output Pin	KIT_LED2	Provides visual feedback	See <a href="#">Figure 5</a>
	KIT_UART_TX	Used for UART transmit (Tx)	See <a href="#">Figure 6</a>
Digital Input Pin	KIT_UART_RX	Used for UART receive (Rx)	See <a href="#">Figure 7</a>

[Figure 2](#) to [Figure 7](#) show non-default configuration settings for the resources.

Figure 2. QSPI Configuration



The screenshot displays the PSoC 6 MCU configuration tool. On the left, the 'Peripherals' pane shows the 'Quad Serial Memory Interface (QSPI) 0' selected. On the right, the 'Parameters' pane for 'Quad Serial Memory Interface (QSPI) 0 (KIT\_QSPI)' is shown. Red boxes highlight the following sections:

- Clocks:**
  - HF Clock: CLK\_HF0 root\_clk [USED]
  - Interface Clock: CLK\_HF2 root\_clk [USED]
  - SPI Clock: P11[7] digital\_inout [USED]
- Data:**
  - SPI Data[0]: P11[6] digital\_out [USED]
  - SPI Data[1]: P11[5] digital\_out [USED]
  - SPI Data[2]: P11[4] digital\_out [USED]
  - SPI Data[3]: P11[3] digital\_out [USED]
  - SPI Data[4]: <unassigned>
  - SPI Data[5]: <unassigned>
  - SPI Data[6]: <unassigned>
  - SPI Data[7]: <unassigned>
- Slave Select:**
  - SPI Slave Select 0: P11[2] digital\_out [USED]
  - SPI Slave Select 1: <unassigned>
  - SPI Slave Select 2: P11[0] digital\_out [USED]
  - SPI Slave Select 3: <unassigned>

Figure 3. F-RAM Memory Configuration

PSoC 6

Slave Slot	Memory Part Number	Data Select	Memory Mapped	Pair With Slot	Start Address	Size	End Address	Write Enable	Config Data In Flash	Encrypt
0	Not used	Quad SPI-Data[0:3]	<input type="checkbox"/>	None	0x18000000	0x10000	0x1800FFFF	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
1	Not used	SPI-MOSI:MISO Data[0:1]	<input type="checkbox"/>	None	0x18010000	0x10000	0x1801FFFF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	CY15B104QSN	SPI-MOSI:MISO Data[0:1]	<input type="checkbox"/>	None	0x18020000	0x10000	0x1802FFFF	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	Not used	SPI-MOSI:MISO Data[0:1]	<input type="checkbox"/>	None	0x18030000	0x10000	0x1803FFFF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Location:

User part number:
 

DEFAULT\_MEMORY

 Erase time:
 

1

 ms

Status register busy mask:
 

0x00

 Chip erase time:
 

16

 ms

Status register quad enable mask:
 

0x00

 Program time (μs):
 

8

Size of memory:
 

0x0000100

 Description:

Program page size:
 

0x0000100

Erase block size (bytes):
 

0x0001000

Number of address bytes for SPI transactions:
 

0x03

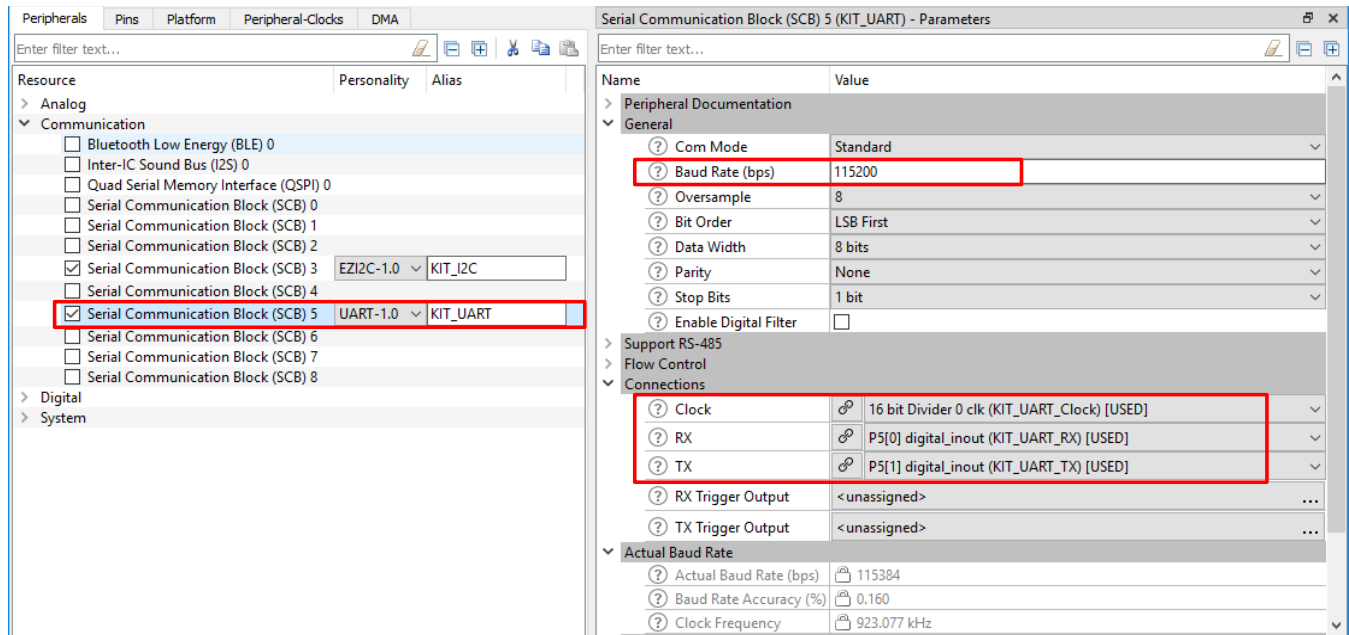
SPI modes supported:
 ☐ Single
 ☐ Dual
 ☒ Quad
 ☐ Octal

Single SPI Commands
 Dual SPI Commands
 Quad SPI Commands
 Octal SPI Commands

Description	Number	Command Width	Address Width	Mode	Mode Width	Dummy Cycles	Data Width
Read command format	0x00	Single	Single	NA	Single	NA	Single
Write enable command format	0x00	Single	Single	NA	Single	NA	Single
Write disable command format	0x00	Single	Single	NA	Single	NA	Single
Erase command format	0x00	Single	Single	NA	Single	NA	Single
Chip erase command format	0x00	Single	Single	NA	Single	NA	Single
Program command format	0x00	Single	Single	NA	Single	NA	Single
Read status register command (containing QE bit)	0x00	Single	Single	NA	Single	NA	Single
Read status register command (containing WIP bit)	0x00	Single	Single	NA	Single	NA	Single
Write status register command (containing QE bit)	0x00	Single	Single	NA	Single	NA	Single

**Note:** Memory mapping of F-RAM (operating in SPI mode) to PSoC 6 MCU's internal memory map is not supported by the QSPI configurator in ModusToolbox 1.1. Memory mapping is enabled by setting CY\_SMIF\_FLAG\_MEMORY\_MAPPED flag in the CY15B104QSN\_SlaveSlot\_2.flags variable. This variable is defined in the generated source file *cycfg\_qspi\_memslot.c*.

Figure 4. UART Configuration



Peripherals Pins Platform Peripheral-Clocks DMA

Serial Communication Block (SCB) 5 (KIT\_UART) - Parameters

Enter filter text...

Resource Personality Alias

> Analog

> Communication

☐ Bluetooth Low Energy (BLE) 0

☐ Inter-IC Sound Bus (I2S) 0

☐ Quad Serial Memory Interface (QSPI) 0

☐ Serial Communication Block (SCB) 0

☐ Serial Communication Block (SCB) 1

☐ Serial Communication Block (SCB) 2

☒ Serial Communication Block (SCB) 3 EZI2C-1.0 KIT\_I2C

☒ Serial Communication Block (SCB) 4

☒ Serial Communication Block (SCB) 5 UART-1.0 KIT\_UART

☐ Serial Communication Block (SCB) 6

☐ Serial Communication Block (SCB) 7

☐ Serial Communication Block (SCB) 8

> Digital

> System

Name Value

> Peripheral Documentation

> General

? Com Mode Standard

? Baud Rate (bps) 115200

? Oversample 8

? Bit Order LSB First

? Data Width 8 bits

? Parity None

? Stop Bits 1 bit

? Enable Digital Filter

> Support RS-485

> Flow Control

> Connections

? Clock 16 bit Divider 0 clk (KIT\_UART\_Clock) [USED]

? RX P5[0] digital\_inout (KIT\_UART\_RX) [USED]

? TX P5[1] digital\_inout (KIT\_UART\_TX) [USED]

? RX Trigger Output <unassigned>

? TX Trigger Output <unassigned>

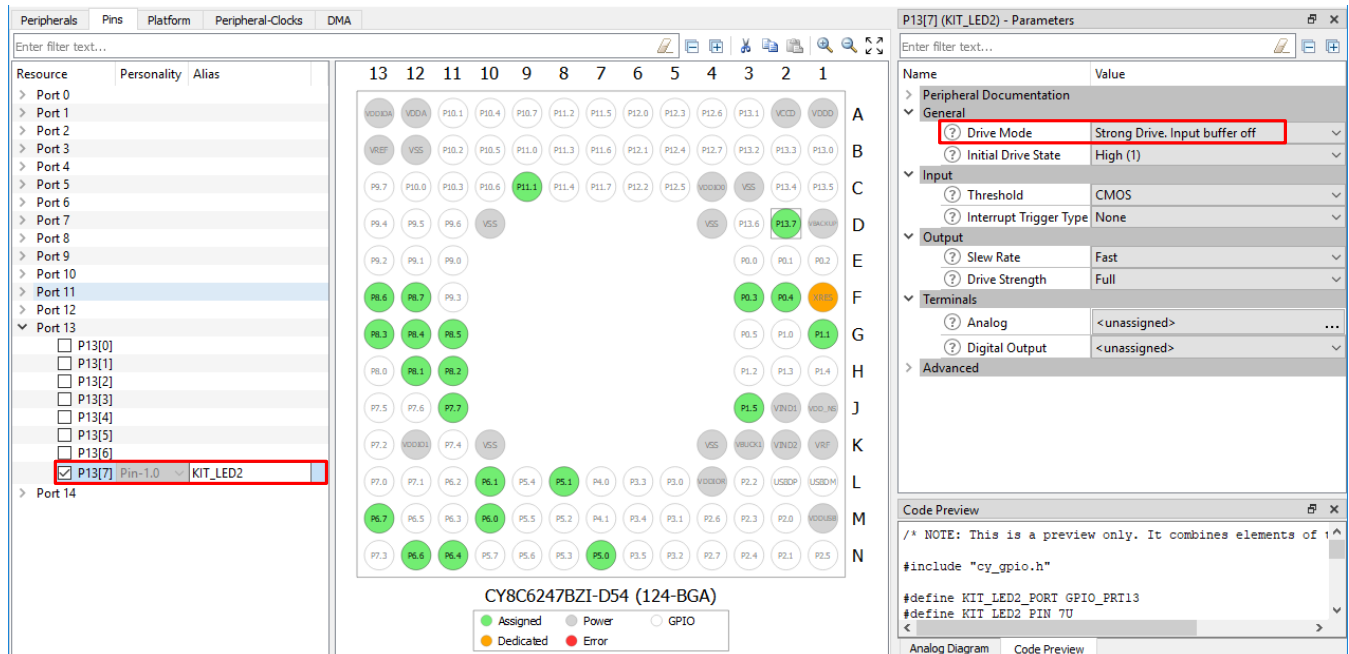
> Actual Baud Rate

? Actual Baud Rate (bps) 115384

? Baud Rate Accuracy (%) 0.160

? Clock Frequency 923.077 kHz

Figure 5. GPIO Pin configuration for LED



Peripherals Pins Platform Peripheral-Clocks DMA

P13[7] (KIT\_LED2) - Parameters

Enter filter text...

Resource Personality Alias

> Port 0

> Port 1

> Port 2

> Port 3

> Port 4

> Port 5

> Port 6

> Port 7

> Port 8

> Port 9

> Port 10

> Port 11

> Port 12

> Port 13

☒ P13[7] Pin-1.0 KIT\_LED2

> Port 14

13 12 11 10 9 8 7 6 5 4 3 2 1

VDDIOA VDDA P10.1 P10.4 P10.7 P11.2 P11.5 P12.0 P12.3 P12.6 P13.1 VDDIOB VDDOB

VREF VSS P10.2 P10.5 P10.8 P11.3 P11.6 P12.1 P12.4 P12.7 P13.2 P13.5

P9.7 P10.0 P10.3 P10.6 P11.1 P11.4 P11.7 P12.2 P12.5 VDDIOX VSS P13.4 P13.5

P9.4 P9.5 P9.6 VSS

VSS P13.6 P13.7

P9.2 P9.1 P9.0

P8.2 P8.1 P8.0

P8.6 P8.7 P8.5

P8.3 P8.4 P8.2

P7.5 P7.6 P7.7

P7.2 VDDIOB P7.4 VSS

P7.0 P7.1 P6.2 P6.1 P5.4 P5.1 P4.0 P3.3 P3.0 VDDIOX P2.2 USBDP USBDM

P6.7 P6.5 P6.3 P6.0 P5.5 P5.2 P4.1 P3.4 P3.1 P2.6 P2.3 P2.0 VDDIOX

P7.3 P6.6 P6.4 P5.7 P5.6 P5.3 P5.0 P3.5 P3.2 P2.7 P2.4 P2.1 P2.5

CY8C6247BZI-D54 (124-BGA)

Assigned Power GPIO Dedicated Error

Name Value

> Peripheral Documentation

> General

? Drive Mode Strong Drive. Input buffer off

? Initial Drive State High (1)

> Input

? Threshold CMOS

? Interrupt Trigger Type None

> Output

? Slew Rate Fast

? Drive Strength Full

> Terminals

? Analog <unassigned>

? Digital Output <unassigned>

> Advanced

Code Preview

/\* NOTE: This is a preview only. It combines elements of

#include "cy\_gpio.h"

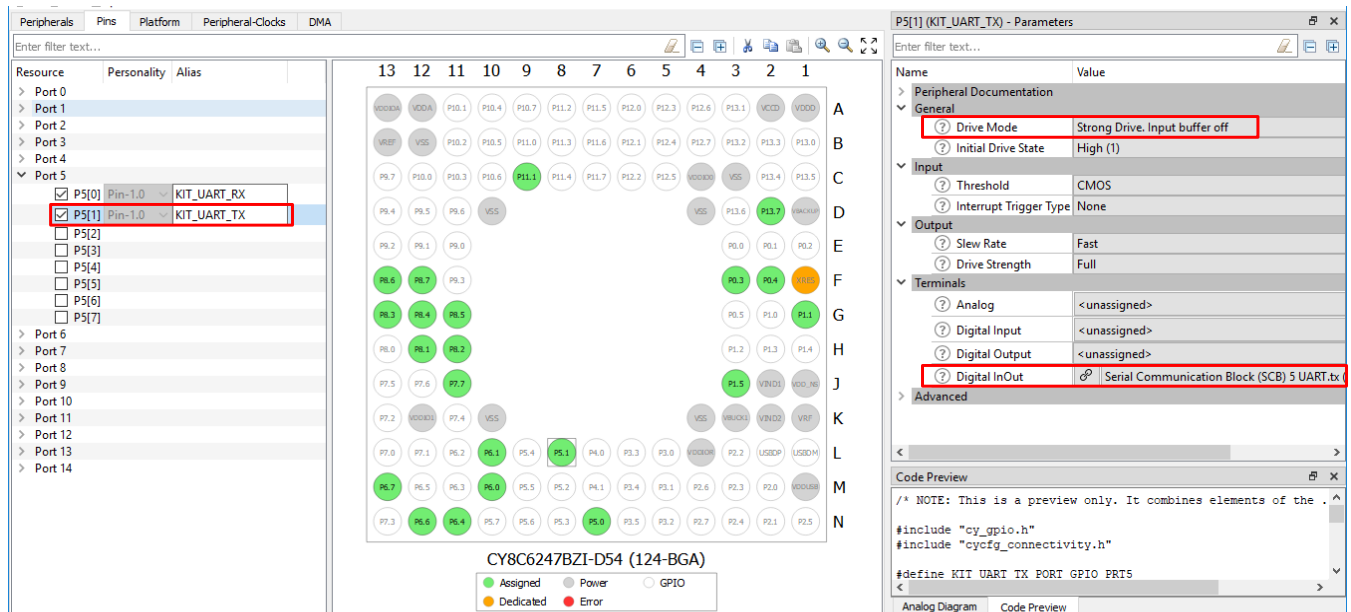
#define KIT\_LED2\_PORT GPIO\_PRT13

#define KIT\_LED2\_PIN 7U

<

Analog Diagram Code Preview

Figure 6. GPIO Pin Configuration for UART Tx



**Peripherals** Pins Platform Peripheral-Clocks DMA

Enter filter text...

Resource Personality Alias

- > Port 0
- > Port 1
- > Port 2
- > Port 3
- > Port 4
- > Port 5
  - ☒ P5[0] Pin-1.0 KIT\_UART\_RX
  - ☒ P5[1] Pin-1.0 KIT\_UART\_TX
  - ☐ P5[2]
  - ☐ P5[3]
  - ☐ P5[4]
  - ☐ P5[5]
  - ☐ P5[6]
  - ☐ P5[7]
- > Port 6
- > Port 7
- > Port 8
- > Port 9
- > Port 10
- > Port 11
- > Port 12
- > Port 13
- > Port 14

**Pin Map (CY8C6247BZI-D54 (124-BGA))**

Legend: ● Assigned ● Power ● Dedicated ● Error

**P5[1] (KIT\_UART\_TX) - Parameters**

Enter filter text...

Name	Value
<b>Peripheral Documentation</b>	
<b>General</b>	
Drive Mode	Strong Drive, Input buffer off
Initial Drive State	High (1)
<b>Input</b>	
Threshold	CMOS
Interrupt Trigger Type	None
<b>Output</b>	
Slew Rate	Fast
Drive Strength	Full
<b>Terminals</b>	
Analog	<unassigned>
Digital Input	<unassigned>
Digital Output	<unassigned>
Digital InOut	Serial Communication Block (SCB) 5 UART.tx
<b>Advanced</b>	

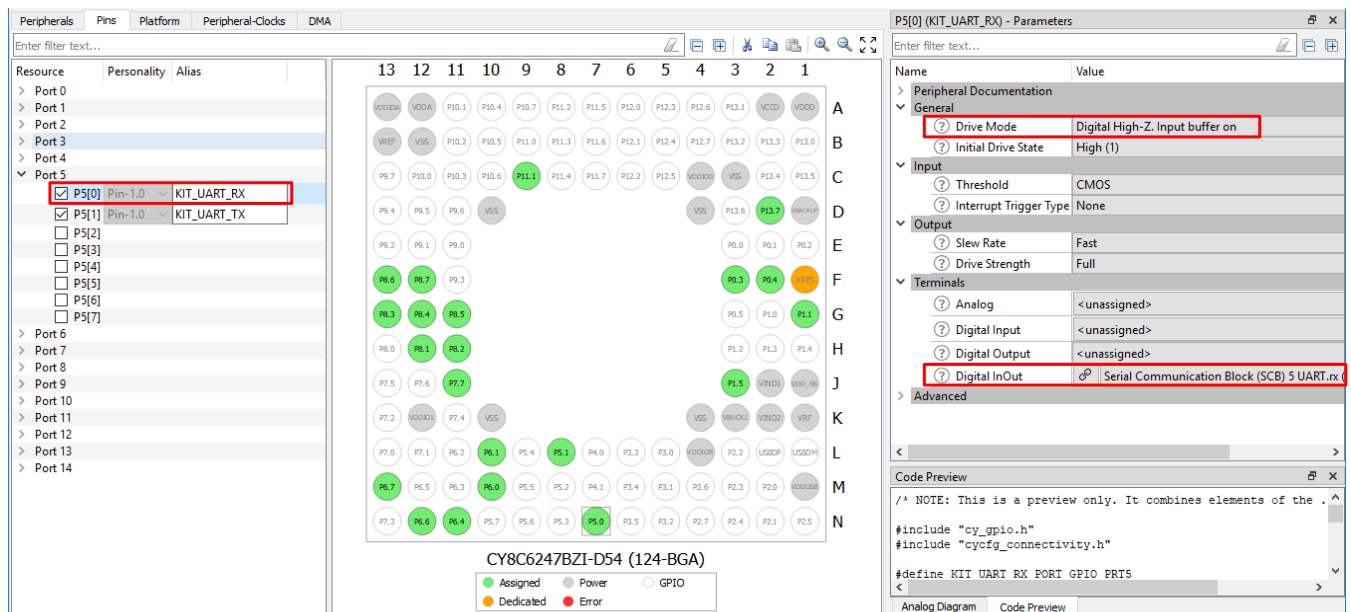
**Code Preview**

```

/* NOTE: This is a preview only. It combines elements of the .
#include "cy_gpio.h"
#include "cycfg_connectivity.h"

#define KIT_UART_TX PORT GPIO PRT5
  
```

Figure 7. GPIO Pin Configuration for UART Rx



**Peripherals** Pins Platform Peripheral-Clocks DMA

Enter filter text...

Resource Personality Alias

- > Port 0
- > Port 1
- > Port 2
- > Port 3
- > Port 4
- > Port 5
  - ☒ P5[0] Pin-1.0 KIT\_UART\_RX
  - ☒ P5[1] Pin-1.0 KIT\_UART\_TX
  - ☐ P5[2]
  - ☐ P5[3]
  - ☐ P5[4]
  - ☐ P5[5]
  - ☐ P5[6]
  - ☐ P5[7]
- > Port 6
- > Port 7
- > Port 8
- > Port 9
- > Port 10
- > Port 11
- > Port 12
- > Port 13
- > Port 14

**Pin Map (CY8C6247BZI-D54 (124-BGA))**

Legend: ● Assigned ● Power ● Dedicated ● Error

**P5[0] (KIT\_UART\_RX) - Parameters**

Enter filter text...

Name	Value
<b>Peripheral Documentation</b>	
<b>General</b>	
Drive Mode	Digital High-Z, Input buffer on
Initial Drive State	High (1)
<b>Input</b>	
Threshold	CMOS
Interrupt Trigger Type	None
<b>Output</b>	
Slew Rate	Fast
Drive Strength	Full
<b>Terminals</b>	
Analog	<unassigned>
Digital Input	<unassigned>
Digital Output	<unassigned>
Digital InOut	Serial Communication Block (SCB) 5 UART.rx
<b>Advanced</b>	

**Code Preview**

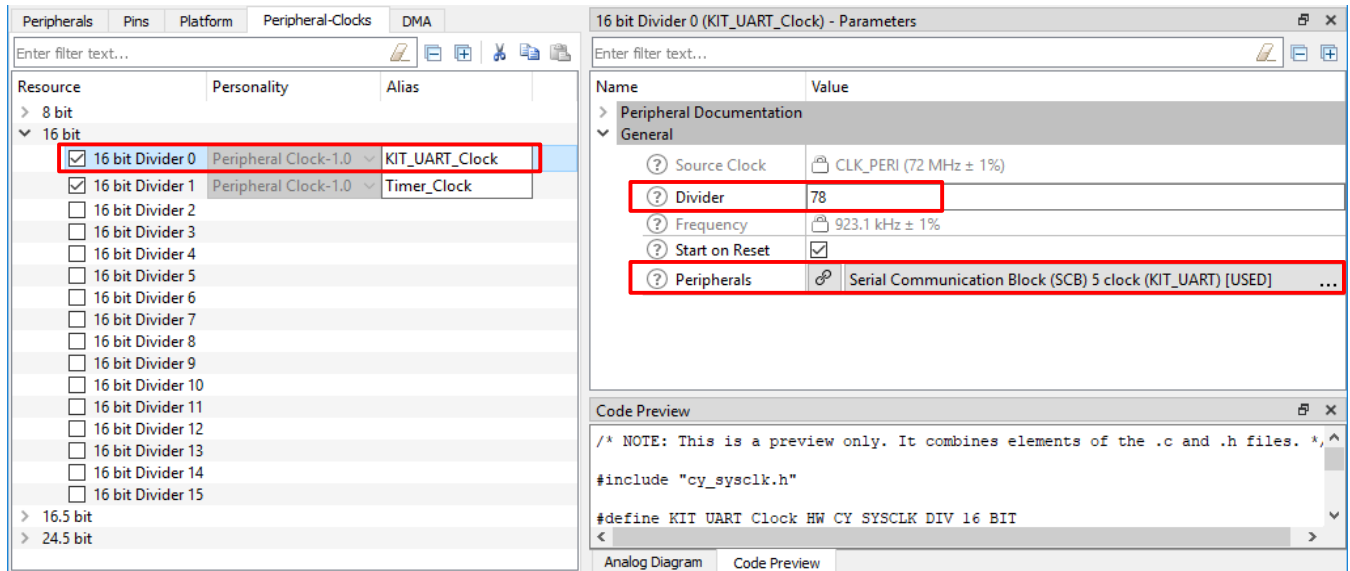
```

/* NOTE: This is a preview only. It combines elements of the .
#include "cy_gpio.h"
#include "cycfg_connectivity.h"

#define KIT_UART_RX PORT GPIO PRT5
  
```

Figure 8 and Figure 9 show the Clock configuration for UART and QSPI resources respectively.

Figure 8. Peripheral-Clock Configuration for UART



16 bit Divider 0 (KIT\_UART\_Clock) - Parameters

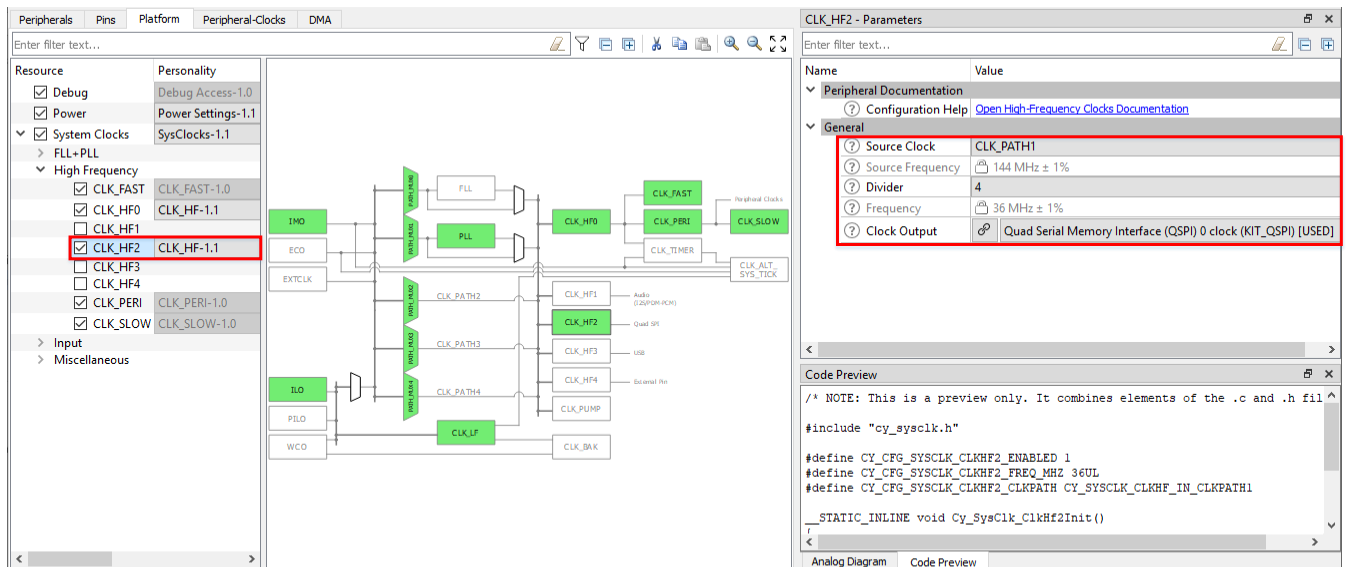
Name	Value
Source Clock	CLK_PERI (72 MHz ± 1%)
Divisor	78
Frequency	923.1 kHz ± 1%
Start on Reset	<input checked="" type="checkbox"/>
Peripherals	Serial Communication Block (SCB) 5 clock (KIT_UART) [USED]

```

/* NOTE: This is a preview only. It combines elements of the .c and .h files. */
#include "cy_sysclk.h"

#define KIT_UART_Clock_HW_CY_SYSClk_DIV_16_BIT
  
```

Figure 9. Clock Configuration for SMIF



CLK\_HF2 - Parameters

Name	Value
Source Clock	CLK_PATH1
Source Frequency	144 MHz ± 1%
Divisor	4
Frequency	36 MHz ± 1%
Clock Output	Quad Serial Memory Interface (QSPI) 0 clock (KIT_QSPI) [USED]

```

/* NOTE: This is a preview only. It combines elements of the .c and .h files. */
#include "cy_sysclk.h"

#define CY_CFG_SYSClk_CLKHF2_ENABLED 1
#define CY_CFG_SYSClk_CLKHF2_FREQ_MHz 36UL
#define CY_CFG_SYSClk_CLKHF2_CLKPATH CY_SYSClk_CLKHF_IN_CLKPATH1

_STATIC_INLINE void Cy_SysClk_ClkHf2Init()
  
```



## Related Documents

For a comprehensive list of PSoC 6 MCU resources, see [KBA223067](#) in the Cypress community.

Application Notes	
<a href="#">AN210781</a> – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 6 MCU with BLE Connectivity devices and how to build your first PSoC Creator project
<a href="#">AN221774</a> – Getting Started with PSoC 6 MCU	Describes PSoC 6 MCU devices and how to build your first ModusToolbox application and PSoC Creator project
Code Examples	
Visit the <a href="#">Cypress GitHub site</a> for a comprehensive collection of code examples using ModusToolbox IDE	
Device Documentation	
<a href="#">PSoC 6 MCU: PSoC 62 Datasheet</a>	<a href="#">PSoC 6 MCU: PSoC 62 Architecture Technical Reference Manual (TRM)</a>
<a href="#">PSoC 6 MCU: PSoC 63 with BLE Datasheet</a>	<a href="#">PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual</a>
Development Kit Documentation	
<a href="#">CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit</a>	
<a href="#">CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit</a>	
<a href="#">CY8CPROTO-062-4343W PSoC 6 Wi-Fi BT Prototyping Kit</a>	
Tool Documentation	
<a href="#">ModusToolbox IDE</a>	The Cypress IDE for IoT designers

## Cypress Resources

Cypress provides a wealth of data at [www.cypress.com](http://www.cypress.com) to help you to select the right device, and quickly and effectively integrate the device into your design.

For the PSoC 6 MCU devices, see [KBA223067](#) in the Cypress community for a comprehensive list of PSoC 6 MCU resources.

## Document History

Document Title: CE227032 – PSoC 6 MCU SMIF On-The-Fly Encryption in MMIO and XIP Modes

Document Number: 002-27032

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6591895	SNVN	06/10/2019	New code example

## 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	<a href="http://cypress.com/arm">cypress.com/arm</a>
Automotive	<a href="http://cypress.com/automotive">cypress.com/automotive</a>
Clocks & Buffers	<a href="http://cypress.com/clocks">cypress.com/clocks</a>
Interface	<a href="http://cypress.com/interface">cypress.com/interface</a>
Internet of Things	<a href="http://cypress.com/iot">cypress.com/iot</a>
Memory	<a href="http://cypress.com/memory">cypress.com/memory</a>
Microcontrollers	<a href="http://cypress.com/mcu">cypress.com/mcu</a>
PSoC	<a href="http://cypress.com/psoc">cypress.com/psoc</a>
Power Management ICs	<a href="http://cypress.com/pmic">cypress.com/pmic</a>
Touch Sensing	<a href="http://cypress.com/touch">cypress.com/touch</a>
USB Controllers	<a href="http://cypress.com/usb">cypress.com/usb</a>
Wireless Connectivity	<a href="http://cypress.com/wireless">cypress.com/wireless</a>

### 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](http://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, 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](http://cypress.com). Other names and brands may be claimed as property of their respective owners.