CE212736 – PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity - Find Me

Objective

This code example demonstrates the implementation of a simple BLE Immediate Alert Service (IAS)-based Find Me Profile (FMP) using PSoC® 6 MCU with Bluetooth Low Energy (BLE) Connectivity, using ModusToolbox™ integrated development environment (IDE).

Requirements

Tool: ModusToolbox IDE 1.1

Programming Language: C

Associated Parts: All PSoC 6 MCU parts with BLE connectivity

Related Hardware: PSoC 6 BLE Pioneer Kit

Overview

This design implements a BLE FMP that consists of an IAS. FMP and IAS are BLE standard Profile and Service respectively, as defined by the Bluetooth SIG. The design uses the RGB LED on the CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit. The blue LED displays the alert level (OFF, flashing, or ON for no alert, mild alert, or high alert respectively). Green and red LEDs indicate whether the Peripheral device (the Pioneer kit) is advertising or disconnected.

The USB-BLE dongle provided with the CY8CKIT-062-BLE Pioneer kit or an iOS/Android mobile device can act as the BLE Central device, which locates the Peripheral device.

Hardware Setup

This example uses the kit's default configuration. Refer to the kit guide to ensure the kit is configured correctly.

Note: The PSoC 6 BLE 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 If you do not upgrade, you will see an error like "unable to find CMSIS-DAP device" or "KitProg firmware is out of date".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 code example consists of two parts: a locator and a target. For the locator, download and install either the CySmart Host Emulation Tool PC application or the CySmart app for iOS or Android. You can test behavior with any of the two options, but the CySmart app is simpler.

Scan the following QR codes from your mobile phone to download the CySmart app.

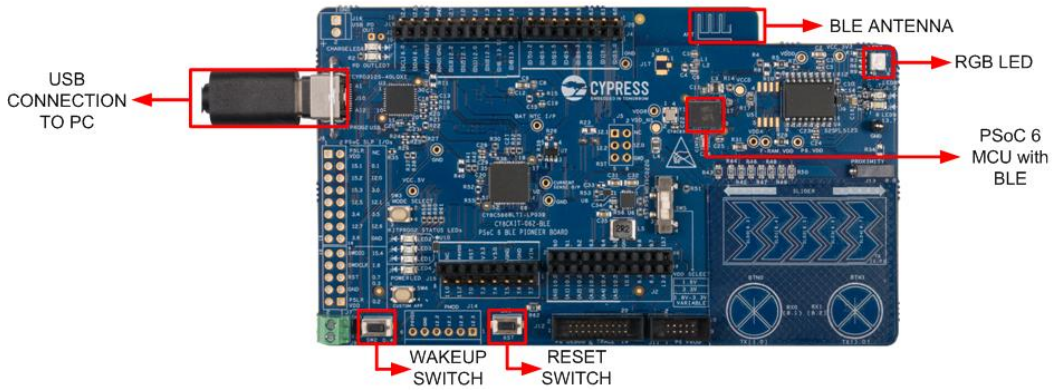
iOS

Android

Operation

1. Connect the kit to your PC using the provided USB cable.

Figure 1. CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit Baseboard



2. Import the code example into a new workspace. If you are not familiar with this process, see KBA225201.
3. Program the PSoC 6 MCU device. In the project explorer, select the **mainapp** project. In the Quick Panel, scroll to the **Launches** section and click the **Program (KitProg3)** configuration.
4. Observe the greed LED turn ON after the device starts advertisement.
5. To test using the CySmart mobile app:
   a. Turn ON Bluetooth on your Android or iOS device.
   b. Launch the CySmart app.
   c. Press the reset switch on the Pioneer Kit to start BLE advertisements from your design. The advertisement LED (green LED) turns ON to indicate that BLE advertisement has started.
   d. Pull down the CySmart app home screen to start scanning for BLE Peripherals; your device appears in the CySmart app home screen. Select your device to establish a BLE connection. Once the connection is established, the green LED turns OFF.
   e. Select the 'Find Me' Profile from the carousel view.
   f. Select an Alert Level value on the Find Me **Profile** screen. Observe the state of the blue LED on the device change based on the alert level.

Figure 2 and Figure 3 show the steps for using CySmart App on iOS and Android respectively.

Figure 2. Testing with the CySmart App on iOS

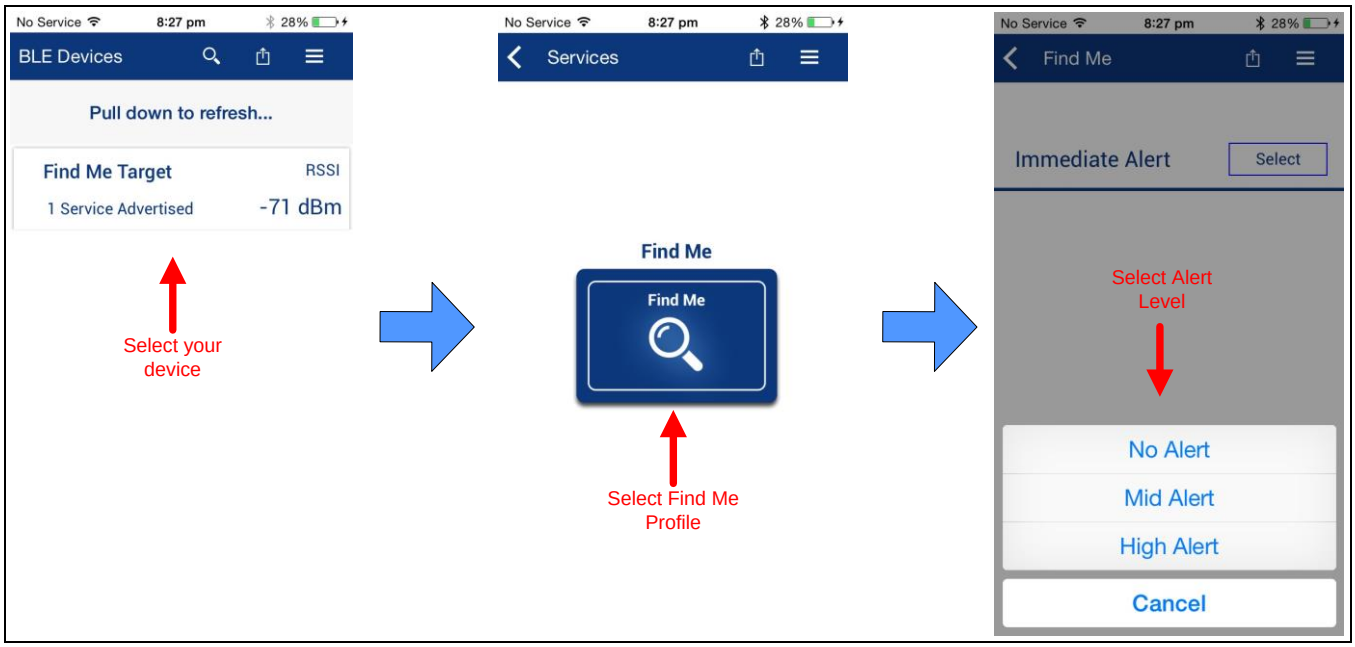
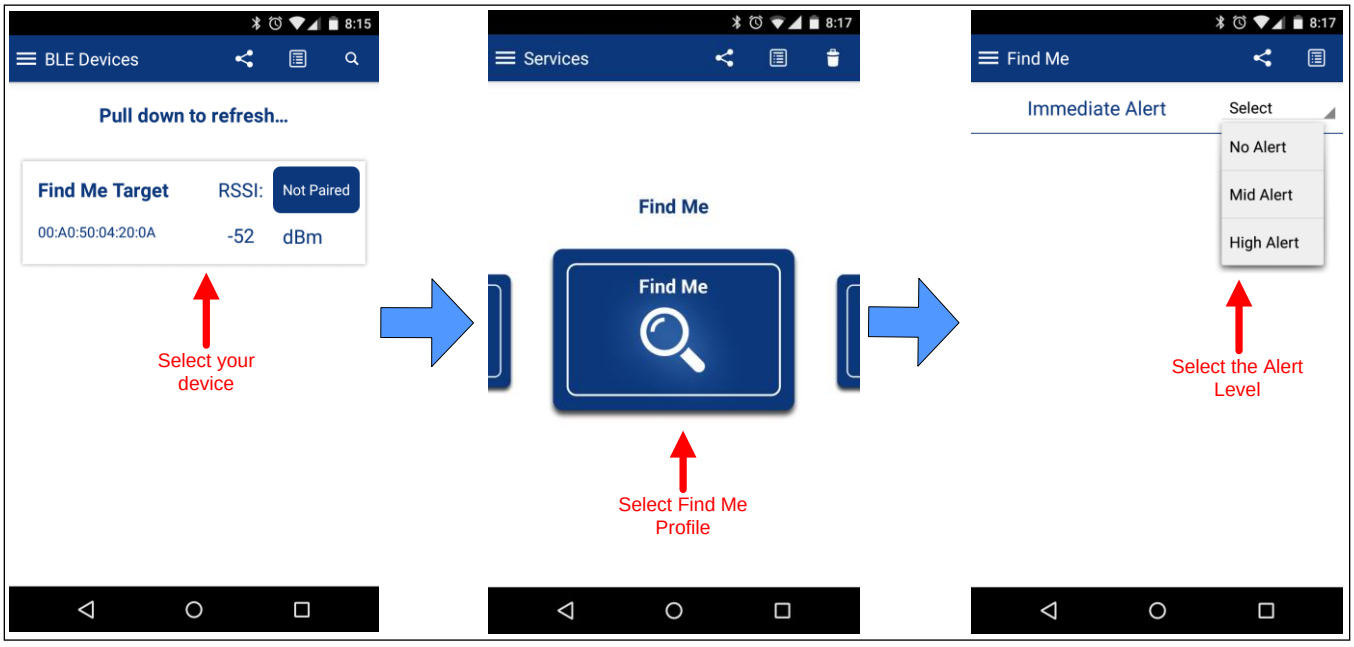


Figure 3. Testing with the CySmart App on Android

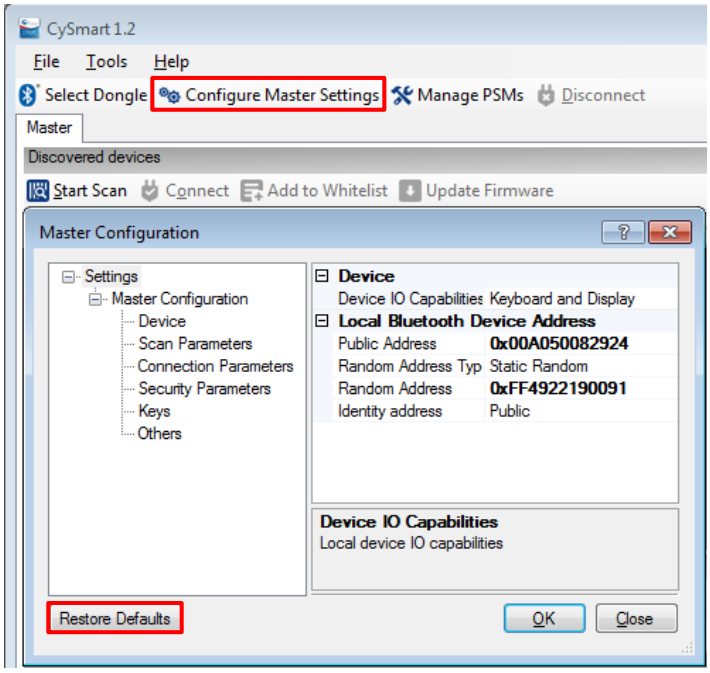


6. To test using the CySmart Host Emulation Tool:
   a. Connect the BLE Dongle to your Windows PC. Wait for the driver installation to complete.
   b. Launch the CySmart Host Emulation Tool.

Note: If the dongle firmware is outdated, you will be alerted. You must upgrade the firmware before you can complete this step. Follow the instructions in the window to update the dongle firmware.

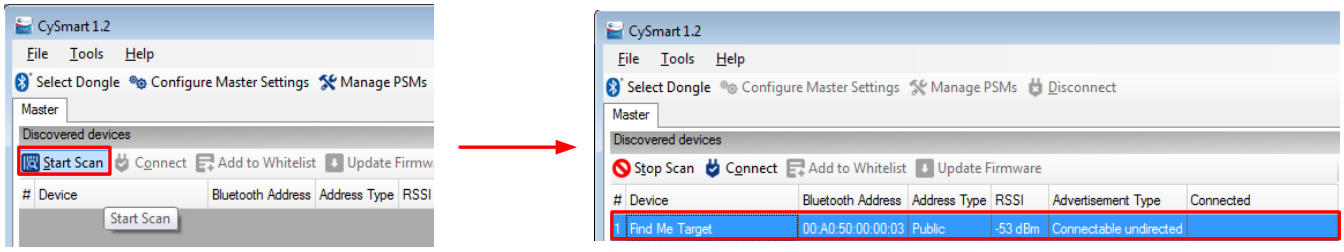
   c. Select **Configure Master Settings** and then, click **Restore Defaults**, as shown in Figure 4. Then, click **OK**.

Figure 4. CySmart Master Settings Configuration



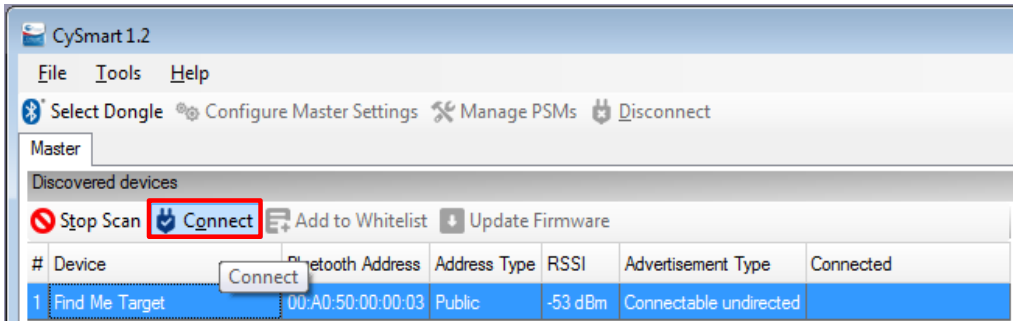
   d. Press the reset switch on the Pioneer Kit to start BLE advertisements from your design.
   e. On the CySmart Host Emulation Tool, click **Start Scan**. Your device name (configured as Find Me Target) should appear in the Discovered devices list, as shown in Figure 5.

Figure 5. CySmart Device Discovery



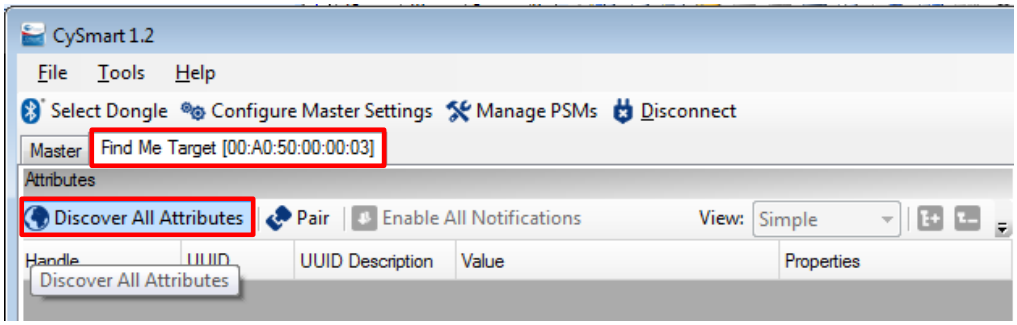
   f. Select your device and click **Connect** to establish a BLE connection between the CySmart Host Emulation Tool and your device, as shown in Figure 6.

Figure 6. CySmart Device Connection



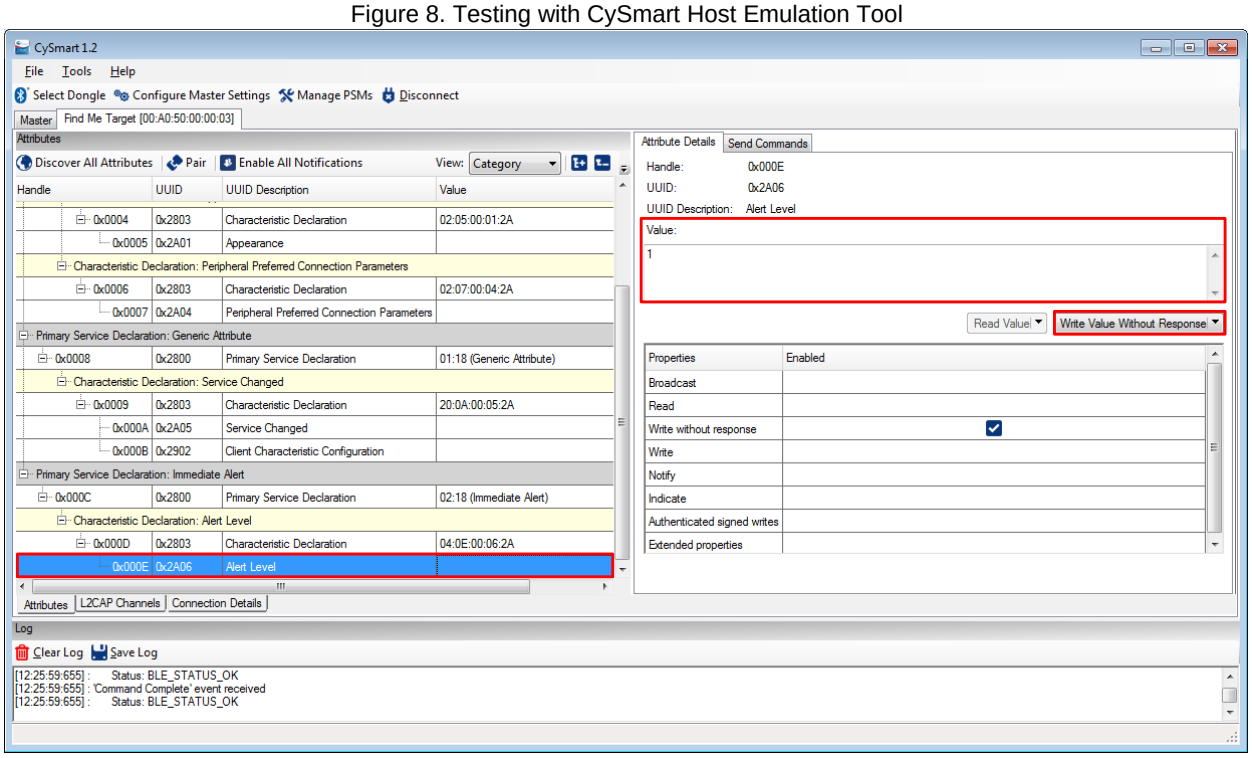
g. Once connected, switch to the **Find Me Target** device tab and discover all Attributes on your design from the CySmart Host Emulation Tool, as shown in Figure 7.

Figure 7. CySmart Attribute Discovery



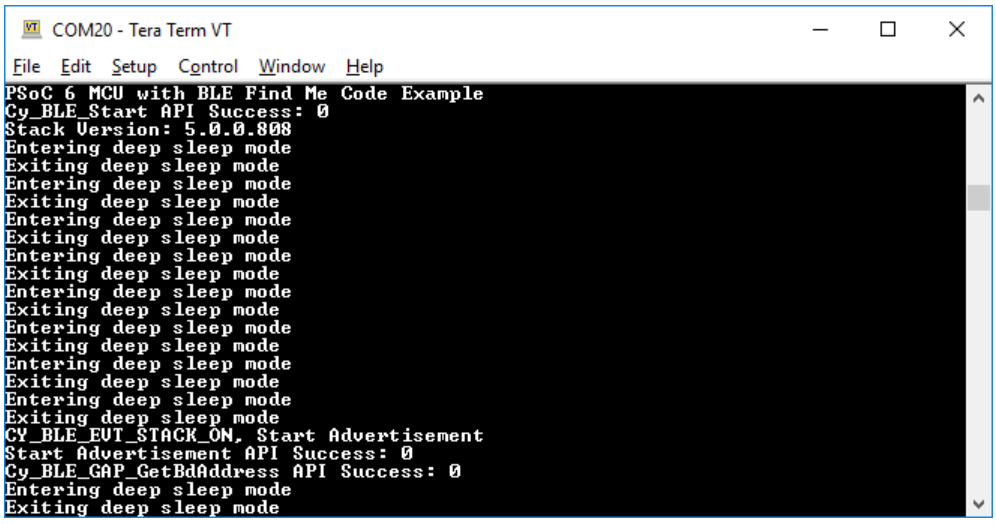
h. Scroll down the **Attributes** window and locate the **Immediate Alert** Service fields. Write a value of 0 – no alert, 1 – mild alert, or 2 – high alert to the **Alert Level** Characteristic under the **Immediate Alert Service**, as Figure 8 shows. Observe that the state of the LED on your device changes per your Alert Level Characteristic configuration.

Figure 8. Testing with CySmart Host Emulation Tool



7. Use the UART debug port to view verbose messages:
   a. The code example ships with the debug port disabled. To enable it, set the macro `DEBUG_UART_ENABLED` in *debug.h* to `ENABLED` and rebuild the code.
   b. Use your favorite serial terminal application and connect to the **KitProg3 USB-UART** COM port. Configure the application to access the COM port at 115200 bps baud rate.
   c. Program the board. The debug messages will appear in the terminal window as shown in Figure 9.

Figure 9. Debug Messages on COM Port



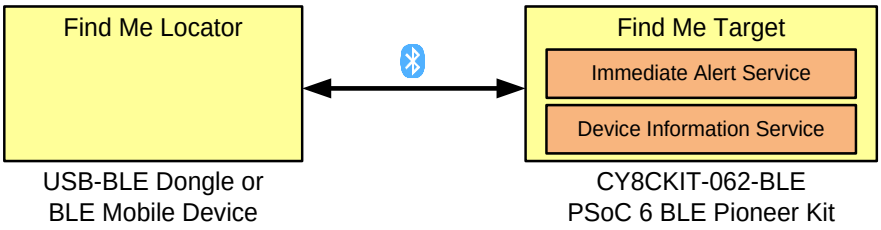
Debugging

You can debug the example to step through the code. Use a **Debug (KitProg3)** configuration. If you are unfamiliar with how to start a debug session with ModusToolbox IDE, see KBA224621 in the Cypress community.

Design and Implementation

The 'Find Me Locator' (the BLE Central device) is a BLE GATT Client. The 'Find Me Target' (the Peripheral device) is a BLE GATT Server with the IAS and an additional Device Information Service implemented, as Figure 10 shows.

Figure 10. Find Me Service Relationship



The BLE Find Me profile defines what happens when the locating Central device broadcasts a change in the alert level.

The Find Me locator performs service discovery using the 'GATT Discover All Primary Services' procedure. The BLE Service Characteristic discovery is done by the 'Discover All Characteristics of a Service' procedure. When the Find Me Locator wants to cause an alert on the Find Me Target, it writes an alert level in the Alert Level Characteristic of the IAS. When the Find Me Target receives an alert level, it indicates the level using the blue LED: OFF for no alert, blinking for mild alert, and ON for high alert.

The BLE interface is implemented on a PSoC 6 MCU with BLE Connectivity device using the BLE resource. The application runs on the Arm® Cortex®-M4 CPU.

See AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity to understand the design of firmware for this code example.

The device enters low-power Deep Sleep mode when BLE is idle. It wakes up automatically when there is activity on the BLE connection.

When BLE is disconnected, the device enters Hibernate mode. It wakes up when the reset switch or wakeup switch (SW2) is pressed and performs a complete reset sequence in firmware.

Resources and Settings

Table 1 lists some of the ModusToolbox resources used in the example, and how they are used in the design. The *design.modus* file contains all configuration settings. For example, for pin usage and configuration, open the **Pins** tab of the design file.

Table 1. ModusToolbox Resources

Resource	Alias	Purpose	Non-default Settings
Bluetooth Low Energy (BLE)	BLE	Implement BLE communication	See Figure 11 through Figure 17
Serial Communication Block (SCB) 5	KIT_UART	Provide a serial interface for verbose messaging	See Figure 18
Multi-Counter Watchdog Timer (MCWDT) 0	MCWDT	Generate a tick every 250 msec	See Figure 19
Digital Output Pin	KIT_RGB_R	Provide visual feedback	See Figure 20
	KIT_RGB_G		
	KIT_RGB_B		
	KIT_UART_TX	Used for UART transmit (Tx)	See Figure 21

Resource	Alias	Purpose	Non-default Settings
Digital Input Pin	KIT_UART_RX	Used for UART receive (Rx)	See Figure 22
	KIT_BTN1	Wake up device from hibernate	See Figure 23

Note: Figure 11 through Figure 23 highlight the non-default settings for each resource in this example. For the clock resource settings, go to the **Platform** tab of the *design.modus* file. Figure 24 shows the middleware configuration for BLE.

Figure 11. BLE: Protocol Configuration

Figure 12. BLE: Adding Find Me Profile

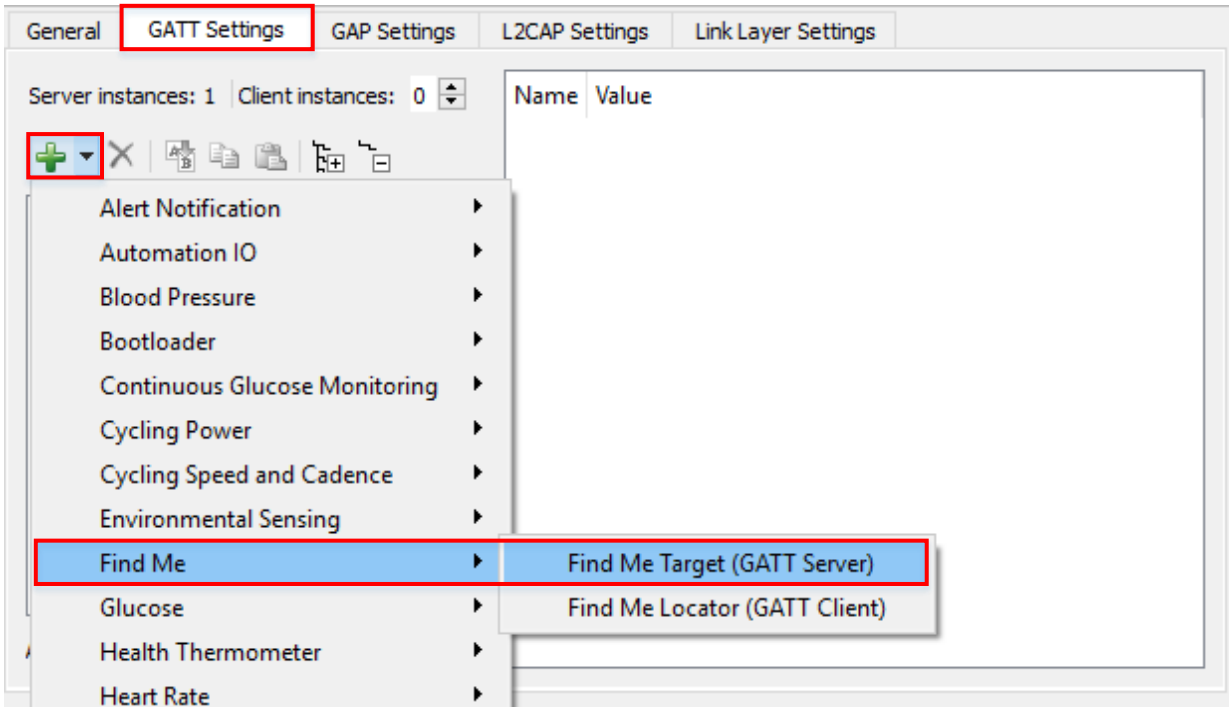


Figure 13. BLE: Device Configuration

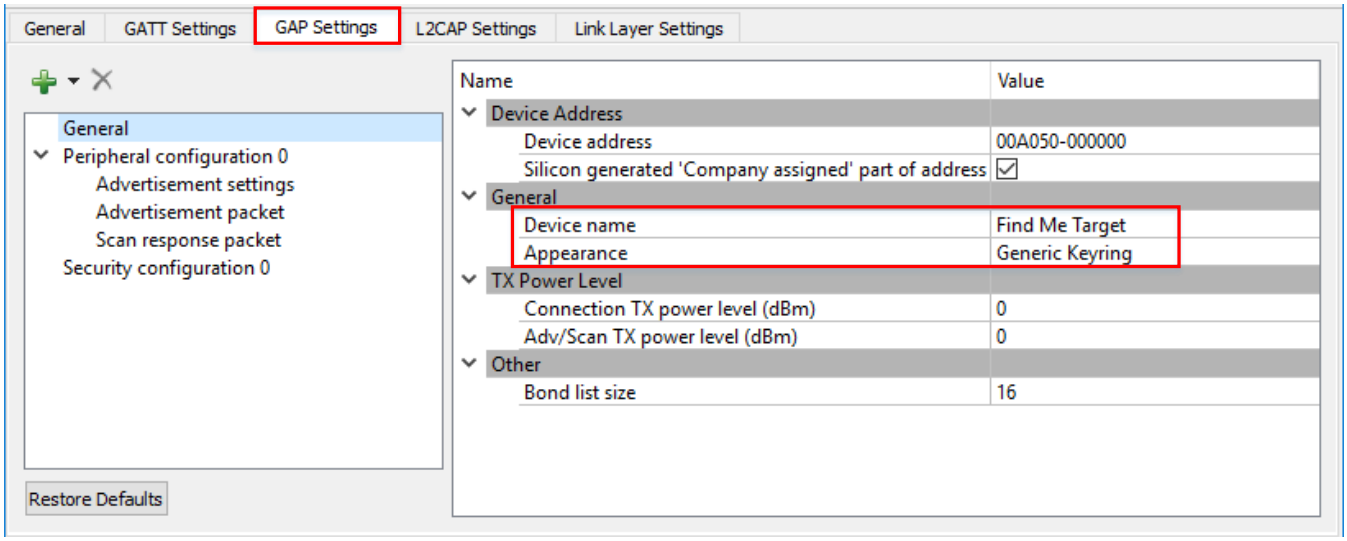


Figure 14. BLE: Advertisement Settings

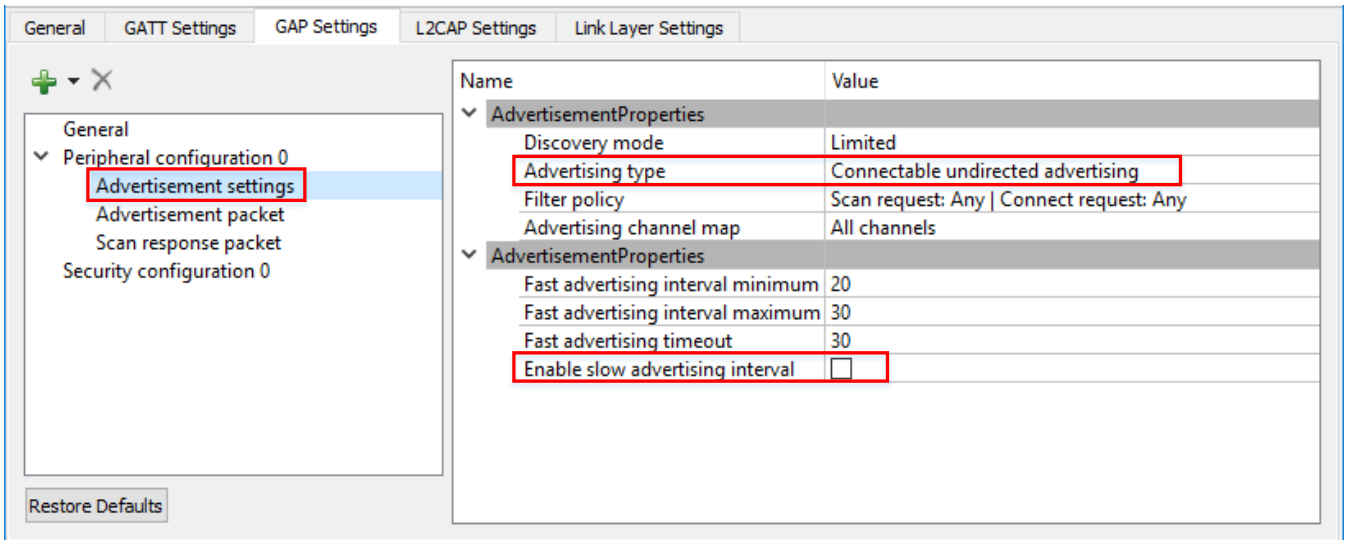


Figure 15. BLE: Advertisement Packet Settings

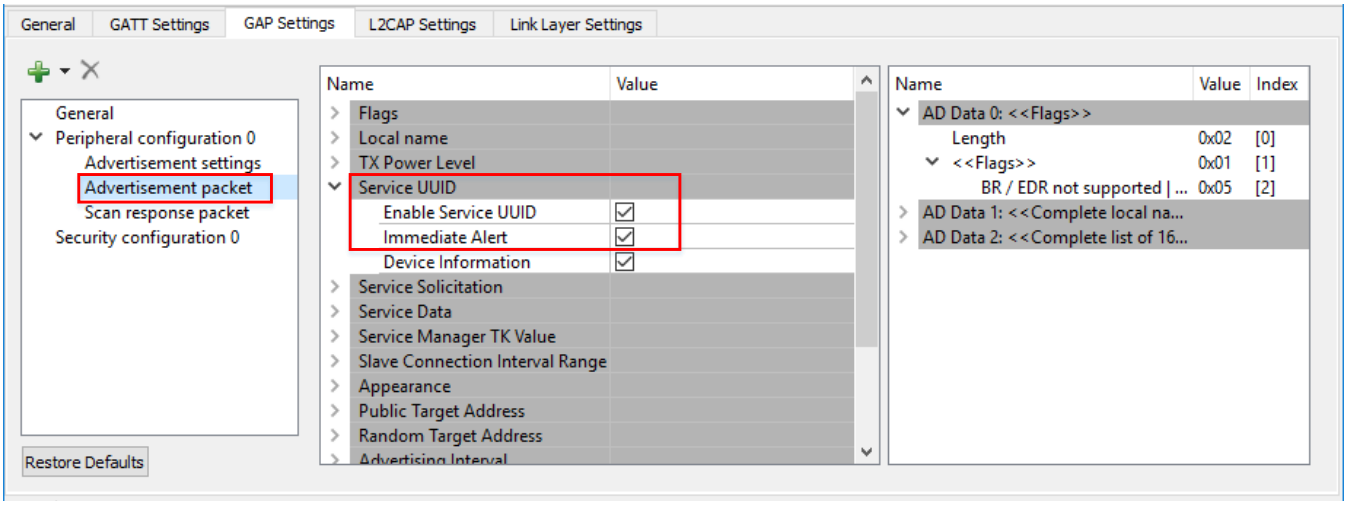


Figure 16. BLE: Response Packet Settings

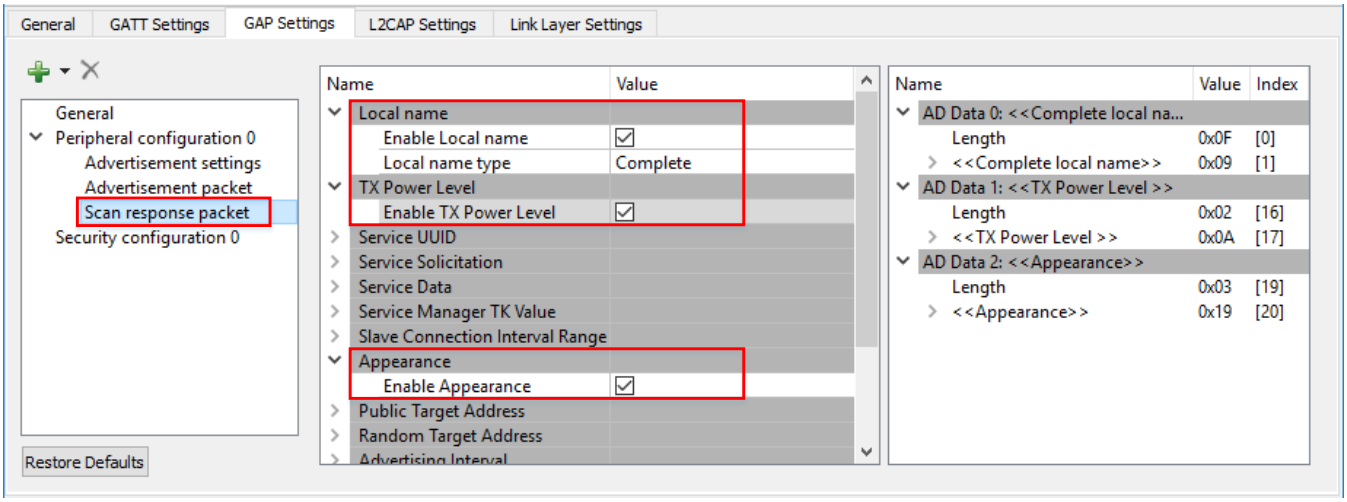


Figure 17. BLE: Security Configuration

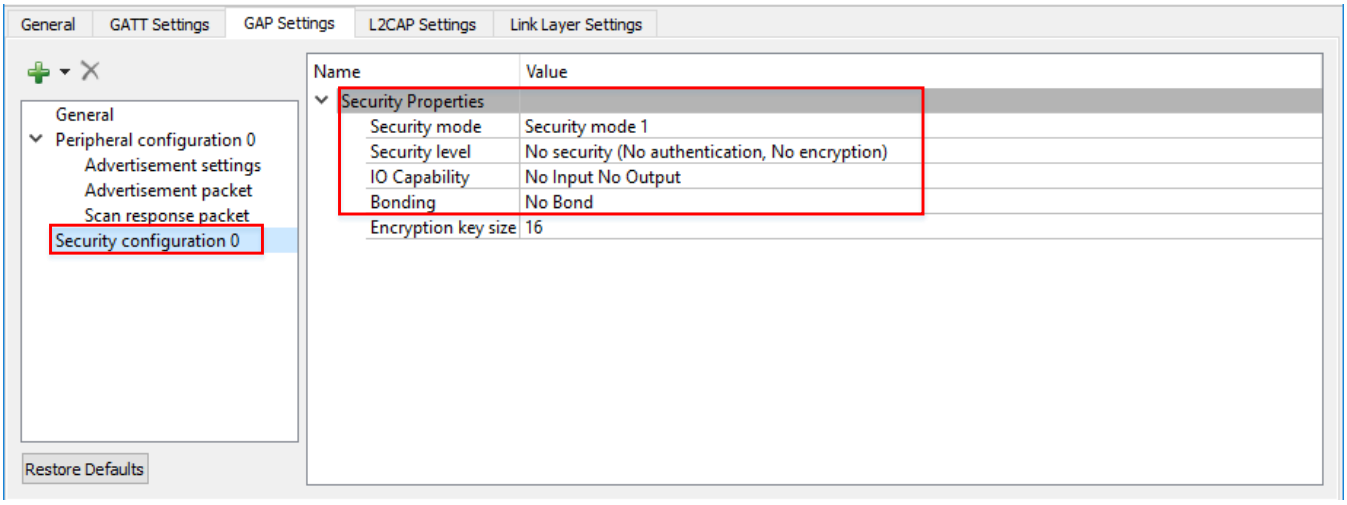


Figure 18. UART Configuration

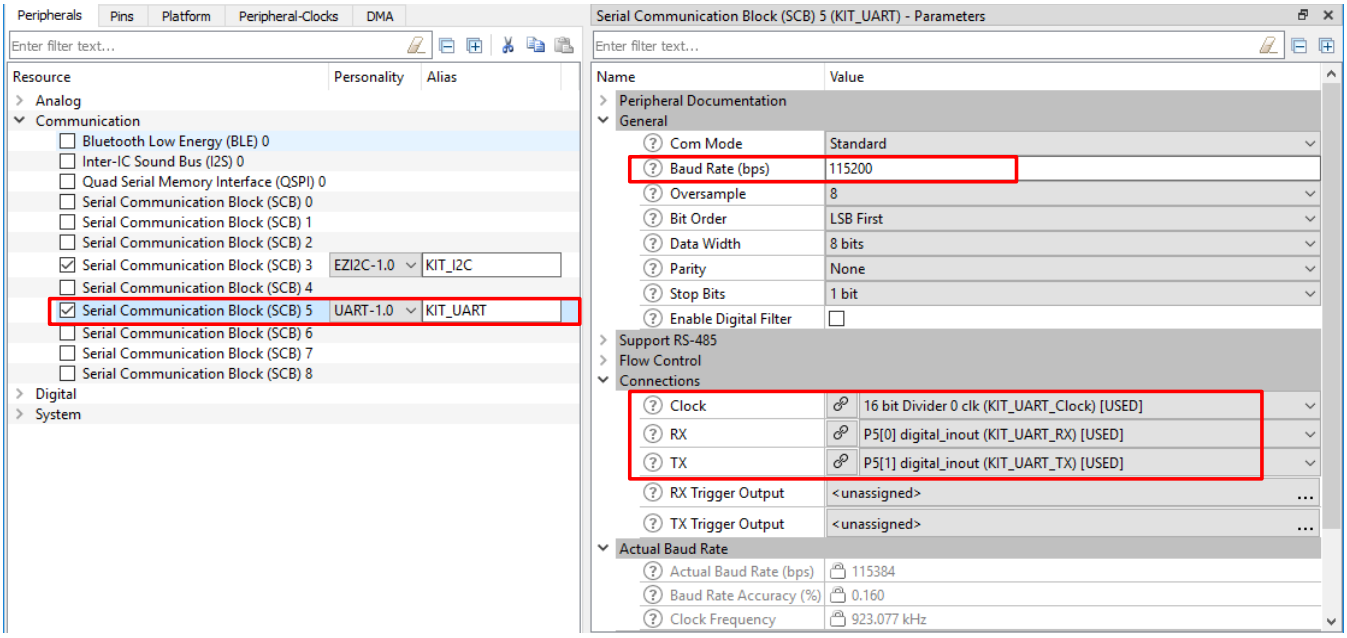


Figure 19. MCWDT Settings

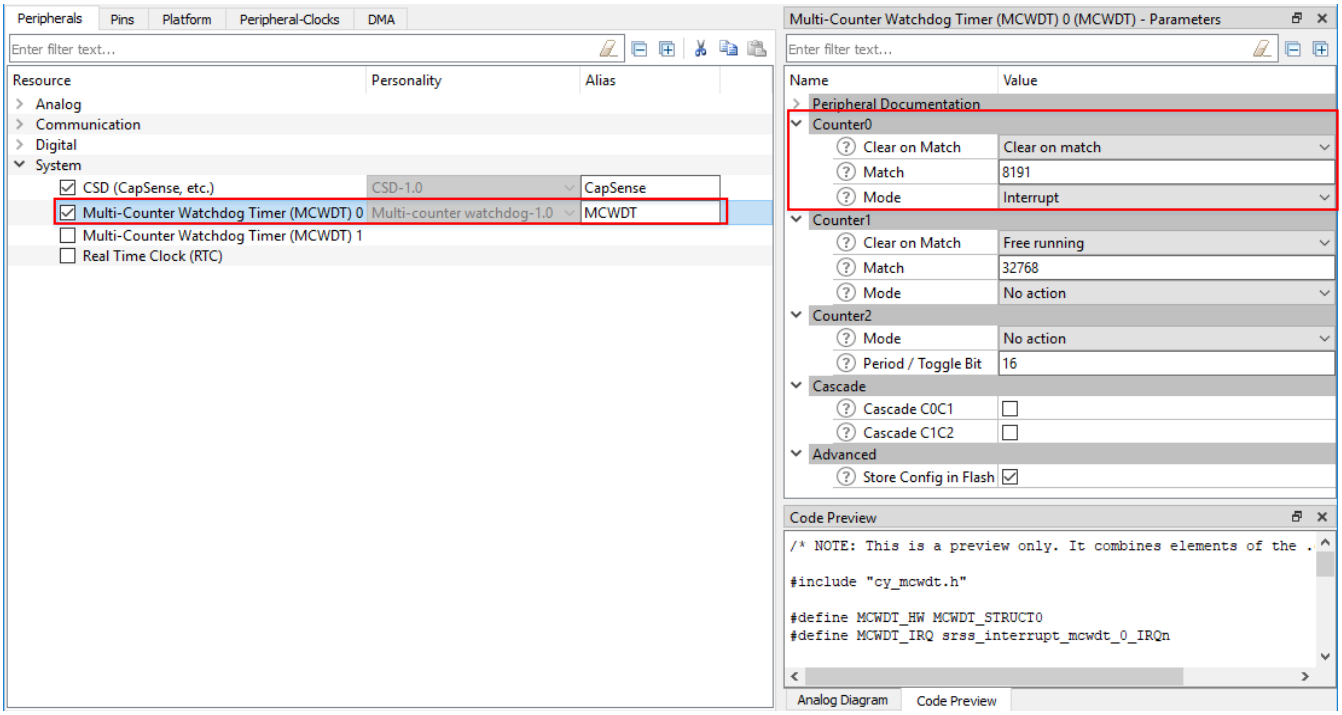


Figure 20. GPIO Pin Configuration for RGB LED

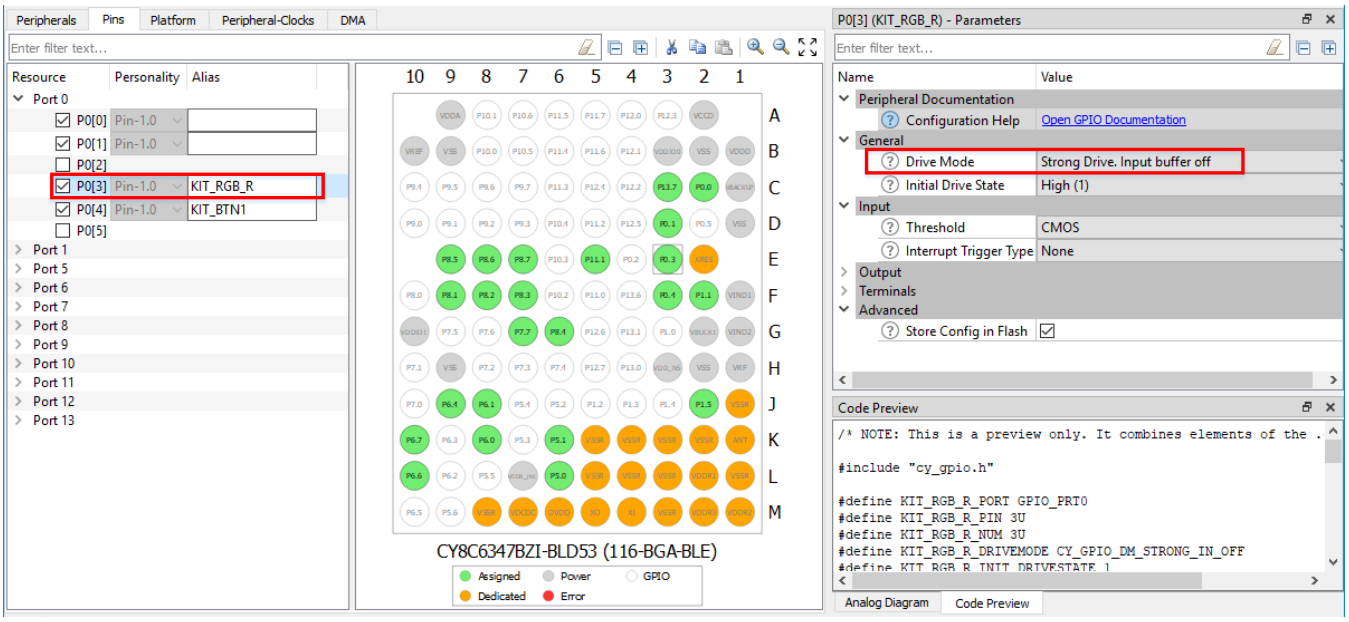


Figure 21. GPIO Pin Configuration for UART Tx

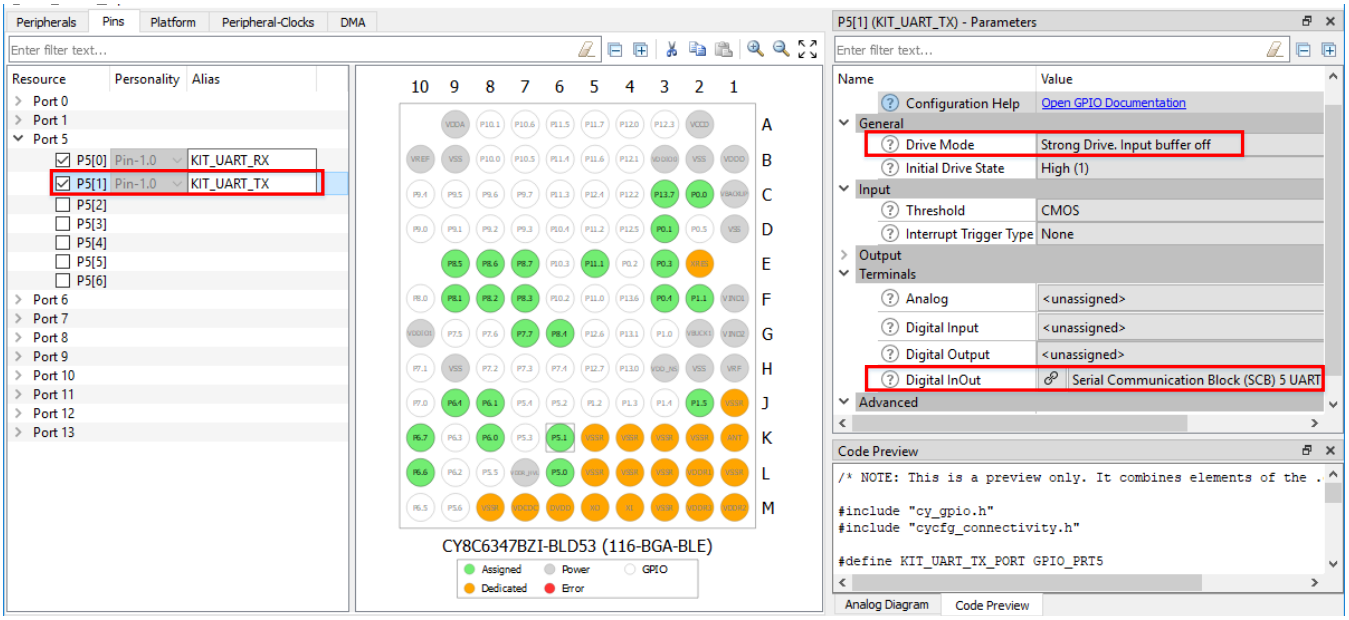


Figure 22. GPIO Pin Configuration for UART Rx

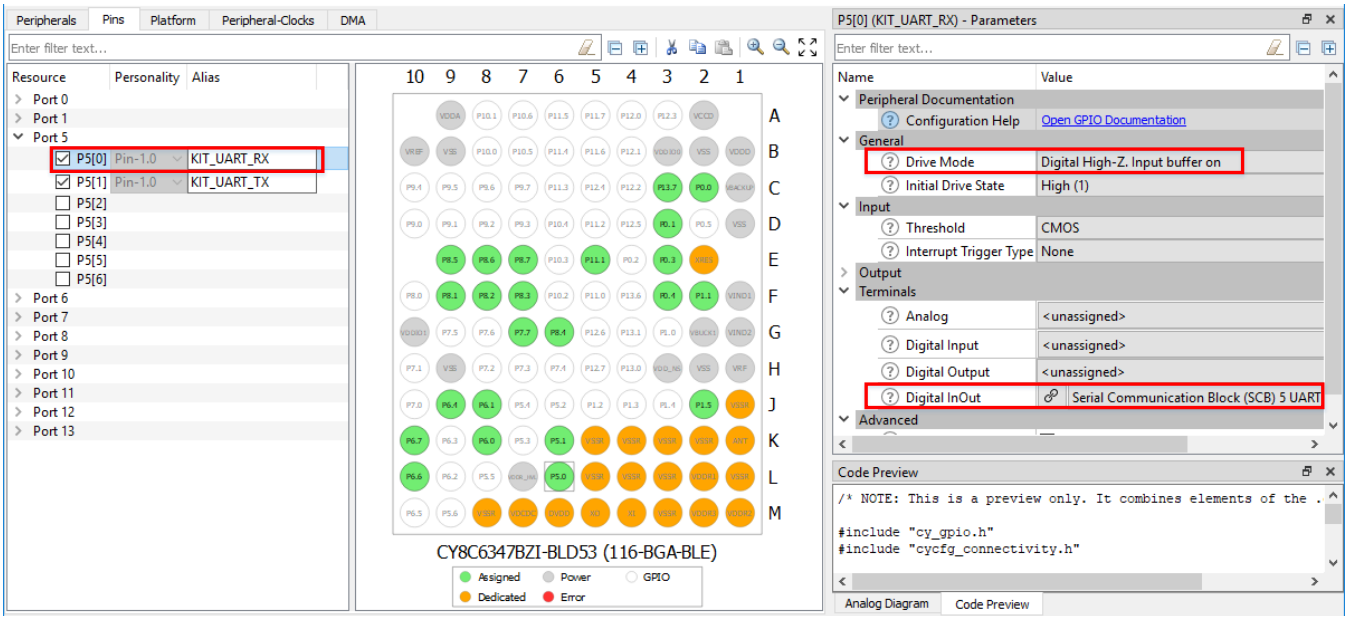


Figure 23. GPIO Pin Configuration for User Button

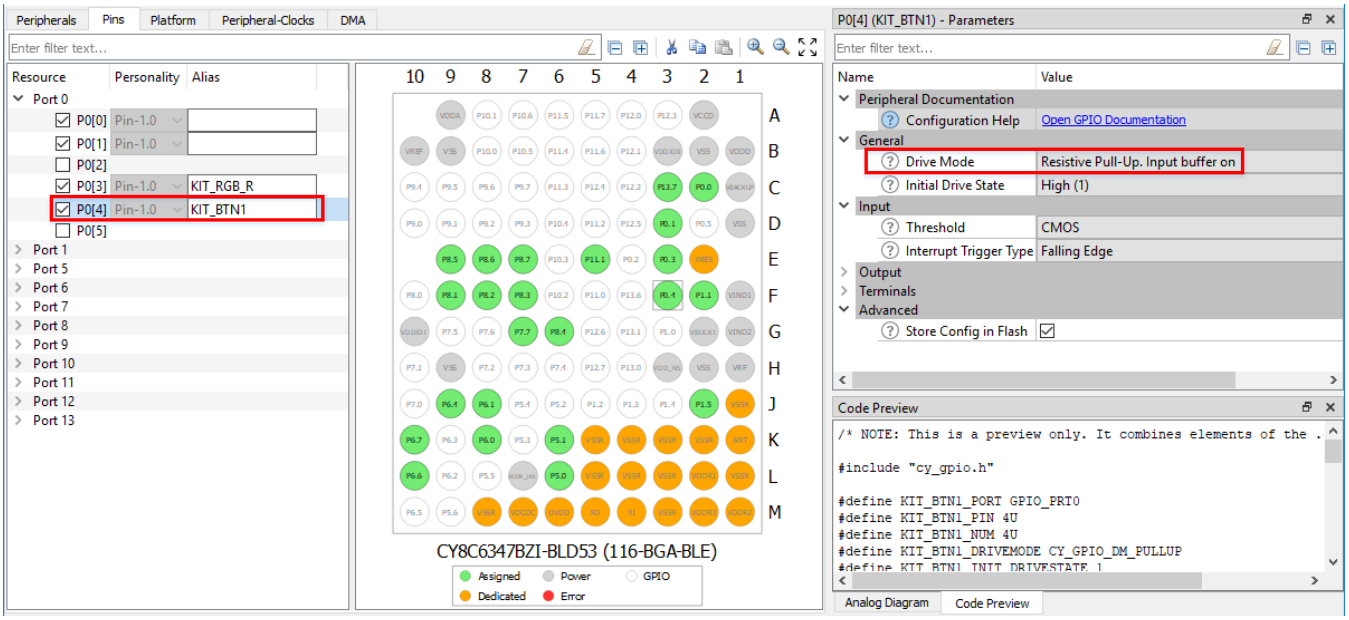
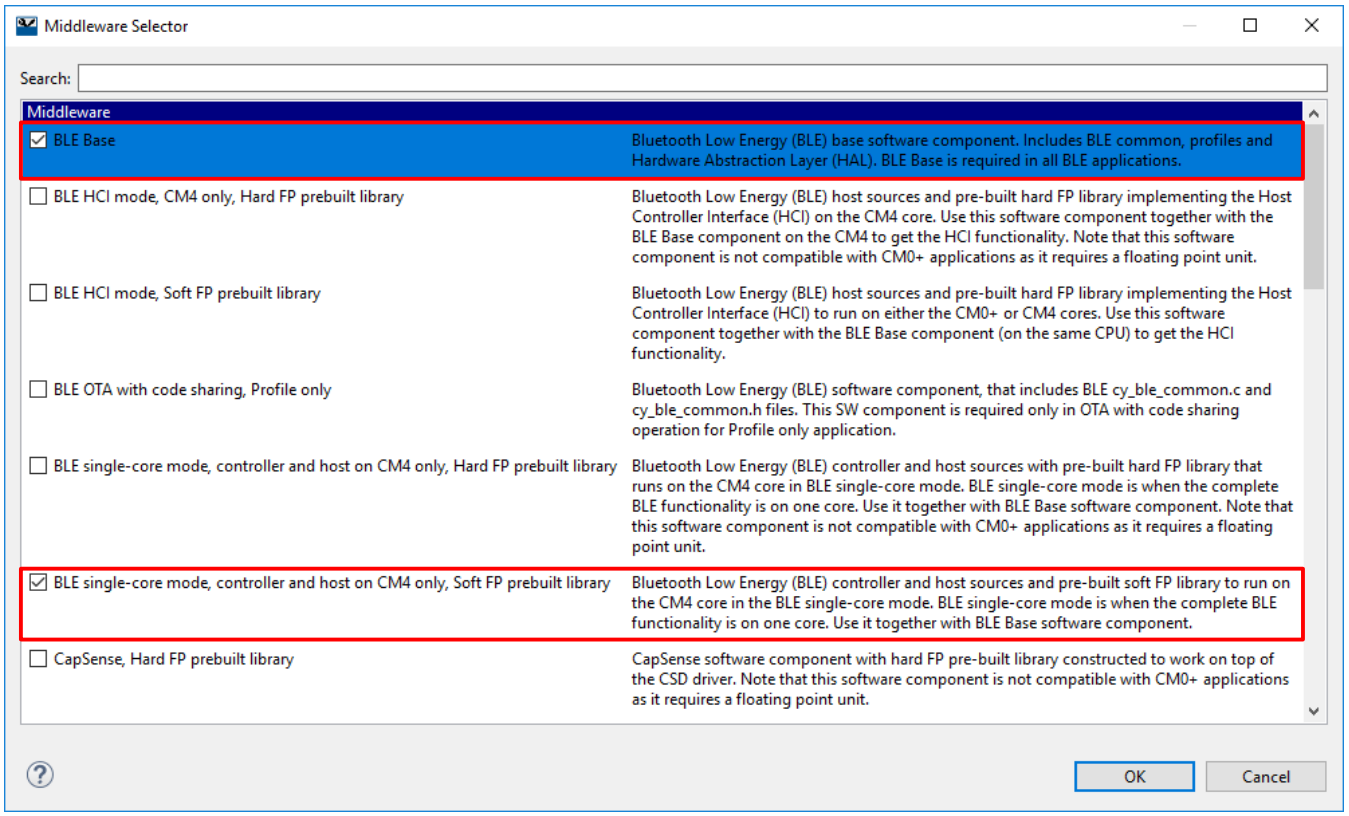


Figure 24. BLE Middleware Configuration



Related Documents

Application Notes	
AN210781 – 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
AN215656 – PSoC 6 MCU: Dual-CPU System Design	Describes the dual-CPU architecture in PSoC 6 MCU, and shows how to build a simple dual-CPU design
Code Examples	
Visit the Cypress GitHub site for a comprehensive collection of code examples using ModusToolbox IDE	
Device Documentation	
PSoC 6 MCU: PSoC 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
Development Kits	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	
CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit	
CY8CPROTO-062-4343W PSoC 6 Wi-Fi BT Prototyping Kit	
Tool Documentation	
ModusToolbox IDE	The Cypress IDE for IoT designers

Cypress Resources

Cypress provides a wealth of data at 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: CE212736 – PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity - Find Me

Document Number: 002-25467

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6366433	SNVN	11/13/2018	New code example
*A	6390655	SNVN	11/21/2018	Added a note about KitProg3 in Hardware Setup
*B	6487573	SNVN	02/18/2019	Code example updated for ModusToolbox 1.1

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

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