CE225909 – PSoC 6 MCU With BLE Connectivity: Battery Level (RTOS)

Objective

This code example demonstrates the implementation of a BLE Battery Service using PSoC® 6 MCU with Bluetooth Low Energy (BLE) Connectivity.

Requirements

Tool: ModusToolbox™ IDE 1.1

Programming Language: C

Associated Parts: All PSoC 6 MCU with BLE Connectivity parts

Related Hardware: PSoC 6 BLE Pioneer Kit

Overview

This code example implements a GATT Server with BLE standard Battery Service and Device Information Service. Battery level is simulated in the firmware; its value changes continuously from 0 to 100 percent. The design uses red LED (LED9) on the CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit for indication (OFF, flashing, or ON for no device connected, advertising, or connected respectively).

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.

This code example uses FreeRTOS. Visit the FreeRTOS website for documentation and API references of FreeRTOS.

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 installed. ModusToolbox works only 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 If you do not upgrade, you will see an error like "unable to find CMSIS-DAP device" or "KitProg firmware is out of date".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 BLE Central device and a BLE Peripheral device. For Central device 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. PSoC 6 MCU is configured as Peripheral.

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

iOS

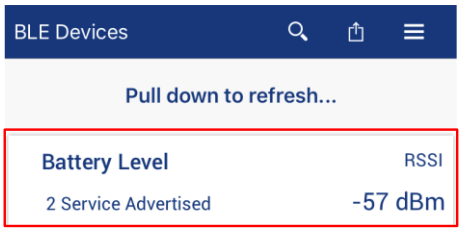
Android

This example uses Tera Term as the terminal emulator.

Operation

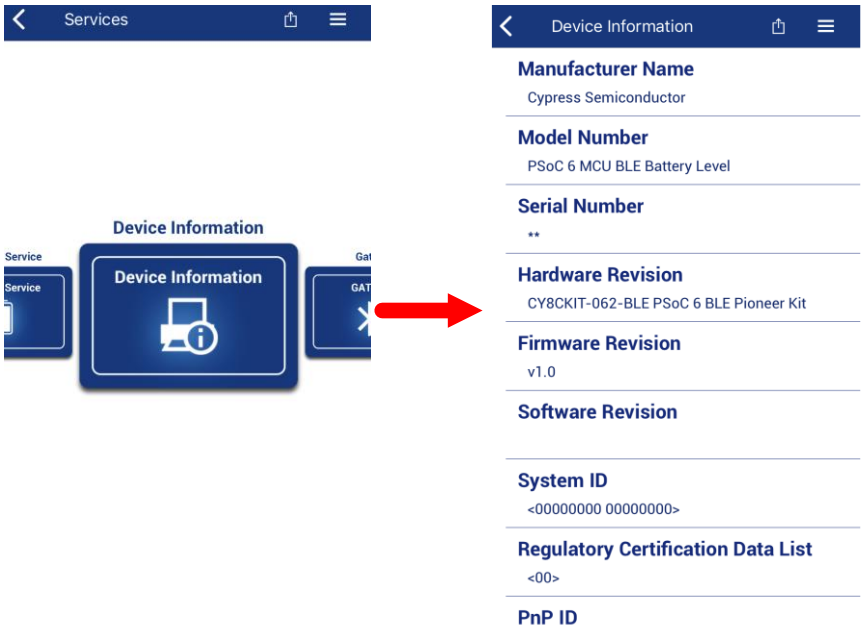
1. Connect the kit to your PC using the provided USB cable.
2. Import the code example to the IDE, in a new workspace. See KBA225201.
3. Program the PSoC 6 MCU device. In the project explorer, select the **mainapp** project. In the Quick Panel, scroll to the **Launches** section and click the **Program (KitProg3)** configuration.
4. To test using the CySmart mobile app:
   a. Turn ON Bluetooth on your Android or iOS device.
   b. Press the reset switch on the Pioneer Kit to start BLE advertisements. The red LED starts blinking to indicate that BLE advertisement has started.
   c. 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 red LED turns ON.

Figure 1. CySmart App Device Discovery



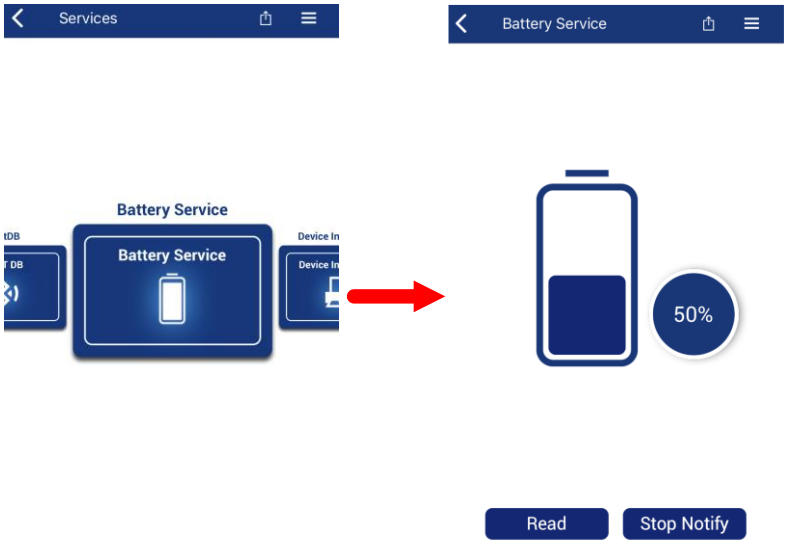
   d. Select the Device Information profile to get the manufacturer, vendor, or both information about the device, as Figure 2 shows.

Figure 2. CySmart App Device Information Service



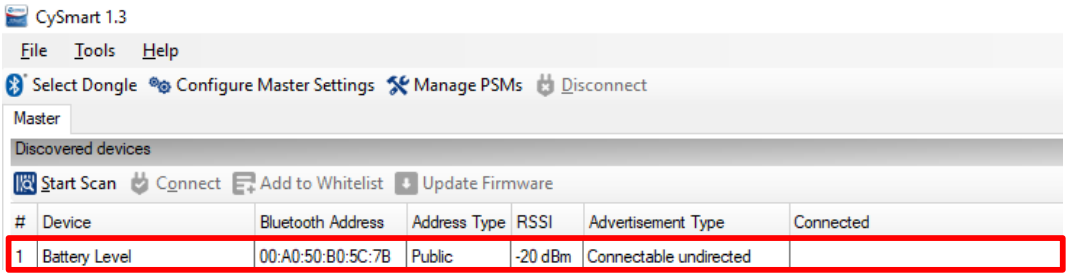
   e. Select **Battery Service** to see the battery level. Tap on **Start Notification** to get notification on every change on battery level.

Figure 3. CySmart App Battery Service



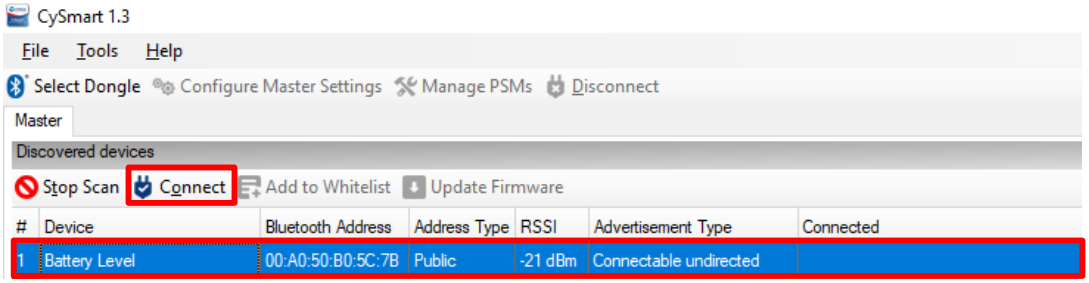
5. To test using the CySmart Host Emulation Tool:
   a. Connect the BLE Dongle to your Windows PC. Wait for the driver installation to complete.
   b. Launch the CySmart Host Emulation Tool.
   c. Press the switch SW2 on the Pioneer Kit to start BLE advertisements from your design. On the CySmart Host Emulation Tool, click **Start Scan**. Your device name (configured as Battery Level) should appear in the Discovered devices list, as shown in Figure 4.

Figure 4. CySmart Device Discovery



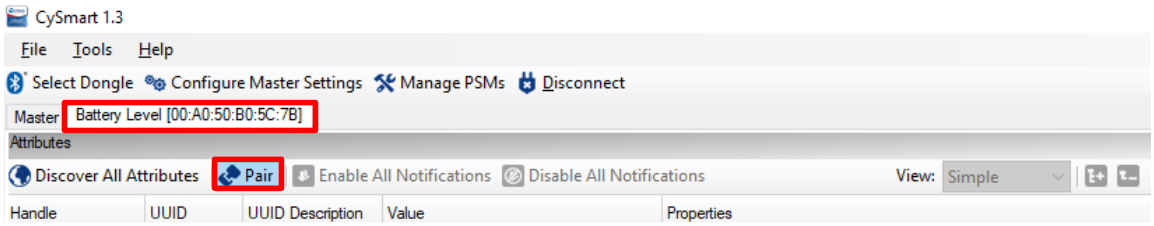
   d. Select your device and click **Connect** to establish a BLE connection between the CySmart Host Emulation Tool and your device, as shown in Figure 5.

Figure 5. CySmart Device Connection



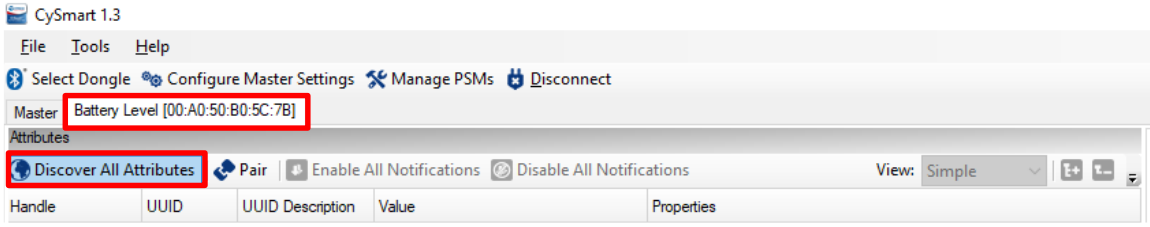
e. Once connected, switch to the **Battery Level** device tab and click on **Pair** as shown in Figure 6.

Figure 6. CySmart Device Paining



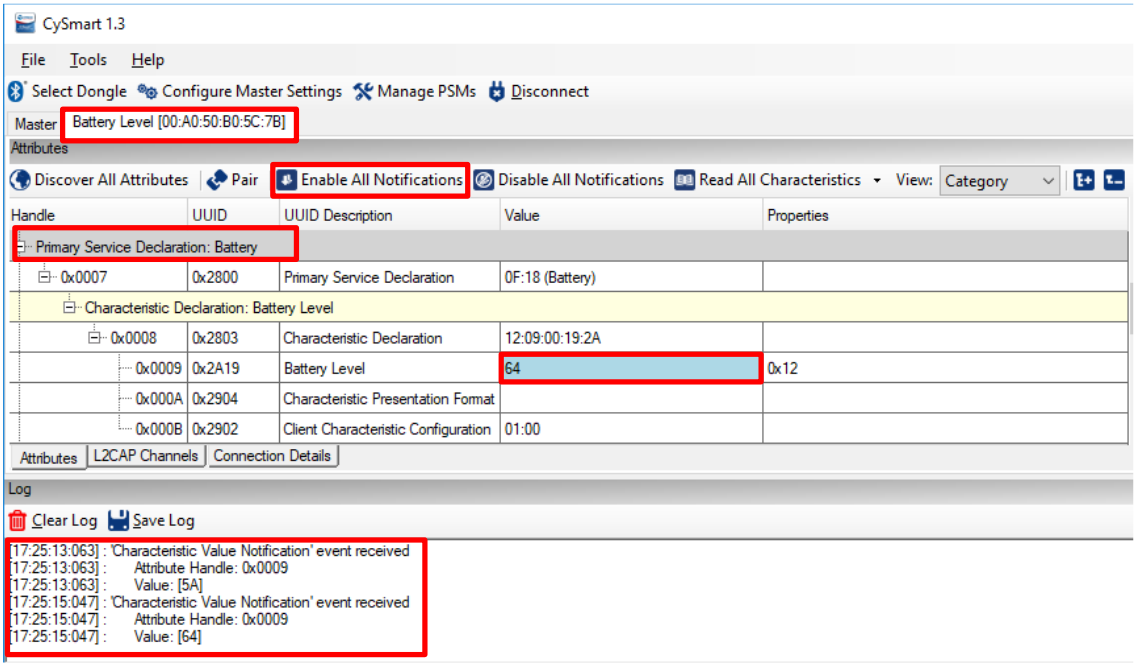
f. Click on **Discover All Attributes** on your design from the CySmart Host Emulation Tool, as shown in Figure 7.

Figure 7. CySmart Attribute Discovery



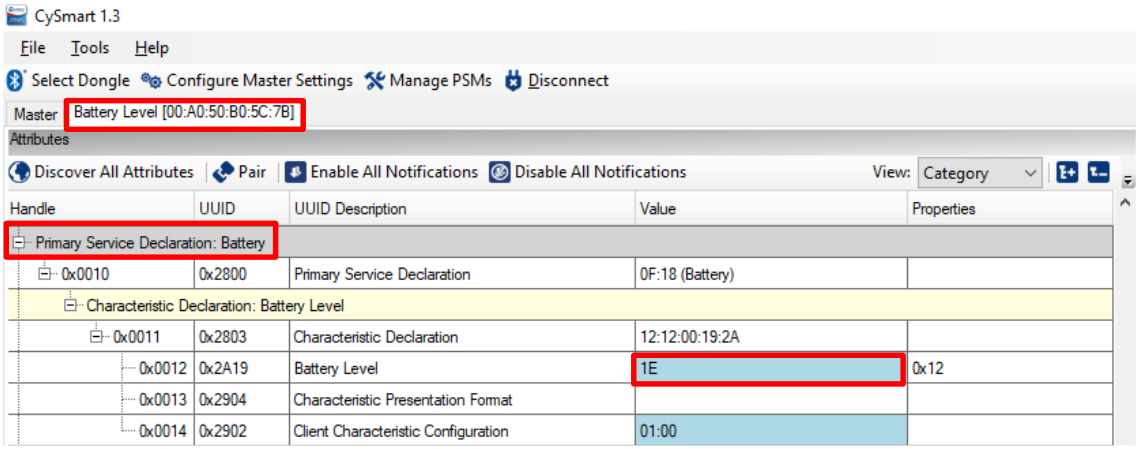
g. Scroll down the **Attributes** window and locate the **Battery** service and then click on **Enable All Notification**. The console must show the battery level as shown in Figure 8.

Figure 8. CySmart Enable All Notification



h. Observe the change in battery level, as Figure 9.

Figure 9. Testing with CySmart Host Emulation Tool



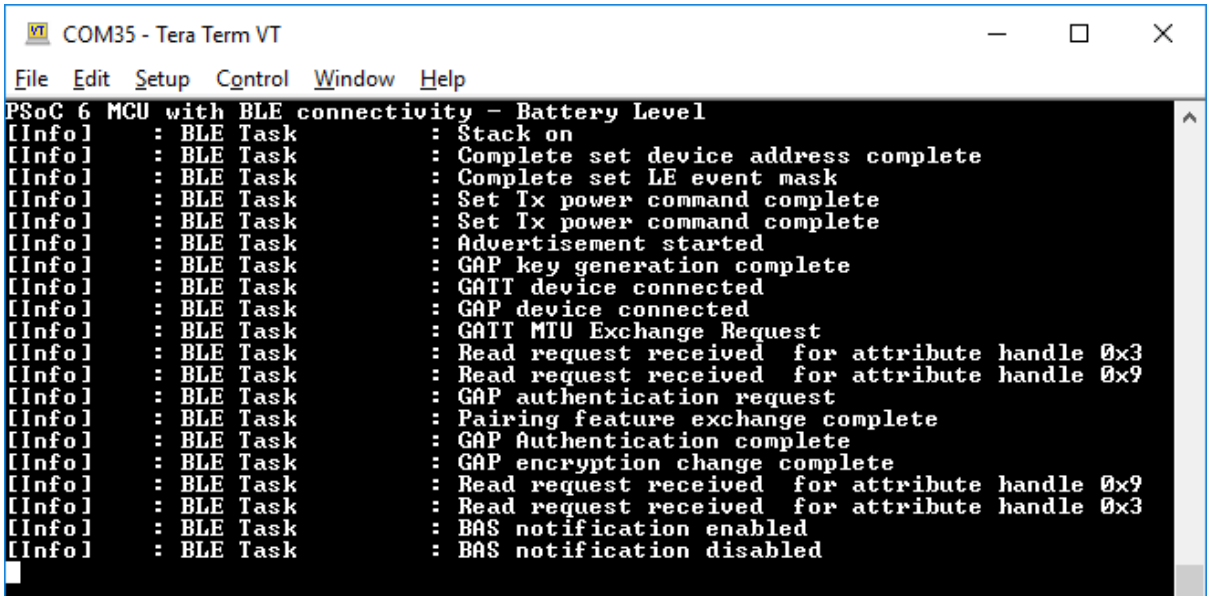
6. Use the UART debug port to view verbose messages:

   a. The code example ships with the debug port disabled. To enable it, set the macro `UART_DEBUG_ENABLE` to `true` in *uart_debug.h* and rebuild the code.

   b. Open your terminal software and select the KitProg COM port, with a baud rate setting of 115200 bps. Set the other serial port parameters to 8N1.

   c. Program the board. The debug messages will appear in the terminal window as shown in Figure 10.

Figure 10. Debug Messages on COM Port



Debugging

You can debug the example to step through the code. Use the **Debug (KitProg3)** configuration. See KBA224621 to learn how to start a debug session with ModusToolbox IDE.

Design and Implementation

In this code example, PSoC 6 BLE is configured in the GAP Peripheral role, and GATT Server with Battery service and Device Information Service is implemented. In this code example, security level is set to “Unauthenticated pairing with encryption”; the device needs to be paired before accessing the Battery Service. If the device is not paired, it can only access the Device Information Service.

The Red LED (LED9) is configured for indication and switch is (SW2) is configured as the advertise switch. A serial communication block (SCB)-based resource in UART mode is configured for printing debug messages.

This application uses FreeRTOS and following RTOS elements are used:

- **Task_Ble**: This task initializes the BLE Host, registers BLE event callbacks, configures the user switch (SW2), and processes the BLE events and commands from other tasks.
- **Task_Battery**: This task is used for simulating battery level.
- **Task_StatusLed**: This task is used for controlling the LED states.
- **Task_Debug**: This task is used for UART-based debug message printing in the terminal application. The code example ships with the debug message printing disabled. To enable it, set the macro `UART_DEBUG_ENABLE` to `true` in *uart_debug.h*.
- Queues (`bleCommandDataQ`, `statusLedDataQ`, `debugMessageQ`) are used for inter-task communications.

Resources and Settings

Table 1 lists some of the PSoC 6 MCU resources used in the example, and how they are used in the design. The *design.modus* file contains all the configuration settings. For example, for pin usage and configuration, open the **Pins** tab of the design file.

Table 1. ModusToolbox Resources

Resource	Alias	Purpose
Bluetooth Low Energy (BLE)	BLE	Implements BLE communication
Serial Communication Block (SCB) 5	KIT_UART	Provides communication between the host and target
Digital Output Pin	KIT_LED	Provides visual feedback
Digital Input Pin	KIT_BTN1	Provides a user interface

Related Documents

Application Notes	
AN221774 - Getting Started with PSoC 6 MCU	Describes PSoC 6 MCU devices and how to build your first ModusToolbox application and PSoC Creator project
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	
CY8CPROTO-063 BLE PSoC 6 BLE Prototyping Kit	
Tool Documentation	
ModusToolbox IDE	ModusToolbox simplifies development for IoT designers. It delivers easy-to-use tools and a familiar microcontroller (MCU) integrated development environment (IDE) for Windows, macOS, and Linux.

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: CE225909 – PSoC 6 MCU With BLE Connectivity: Battery Level (RTOS)

Document Number: 002-25909

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6426296	AJYA	03/27/2019	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	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 | Code Examples | Projects | Videos | Blogs | Training| Components

Technical Support

cypress.com/support

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