CE217645 - BLE Weight Scale Profile with PSoC 6 MCU with BLE Connectivity

Objective

This example demonstrates how to configure and use BLE Component API functions and application layer callback for the BLE Weight Scale application.

Overview

The design demonstrates the Weight Scale Profile operation of the BLE Component. The Weight Scale Sensor uses one instance of the Weight Scale Service (WSS), Body Composition Service (BCS), User Data Service (UDS), and Device Information Service (DIS) to simulate weight measurements for up to four registered users. The Weight Scale Sensor operates with other devices that implement the Weight Scale Collector Profile.

Requirements

Tool: PSoC Creator™ 4.2

Programming Language: C (Arm® GCC 5.4-2016-q2-update)

Associated Parts: All PSoC 6 BLE parts

Related Hardware: CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit

Hardware Setup

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

1. Connect the BLE Pioneer Kit to the computer's USB port.
2. Connect the BLE Dongle to one of the USB ports on the computer.

LED Behavior for VDDD Voltage < 2.7 volts

If the VDDD voltage is set to less than 2.7 volts in the DWR settings **System** tab, only the red LED is used. The red LED blinks to indicate that the device is advertising. The red LED is OFF when the device is connected to a peer device. When the device is in Hibernate mode, the red LED stays ON.

Software Setup

BLE Host Emulation Tool

This example requires the CySmart application. 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 one of the following QR codes from your mobile phone to download the CySmart app.

iOS

Android

Terminal Tool

This example uses a terminal window. You must have terminal software, such as Tera Term, or PuTTy.

Operation

The Weight Scale uses several BLE Services in its operation, such as Weight Scale, User Data, Body Composition, and Device Information. For simplicity, the simulation of Body Composition measurements is not implemented in the example project. The Weight Scale is configured to generate new weight measurements for the currently active user every seven seconds. Measurements are sent with notifications. The measurement data includes Flags, Weight, Height and BMI. Simulation starts from the value of 70 kg (the project is configured to send only metric values) and is incremented by 0.5 kg every 7 seconds. When the weight reaches 80 kg, the simulation is reset back to 70 kg.

The User Data Service is used for managing different user records – up to four records in the current example. Initially, the project has only one registered user, so create the other three user records if required. A new user can be created – send the **Register New User** command to the User Control Point. The UDS also supports the **Consent** and **Delete User** commands. Refer to the UDS specification for a detailed description and the commands format.

Registering New User

Initially, the project has only one registered user. The user record has the following default values:

First name – John
Last Name – Smith
Age – 25
Gender – Male
Weight – 70 kg (14000 with resolution 0.005 kg)
Height – 1.7 m (1700 with resolution 0.01 kg)

All newly registered users' records will be initialized with these default values. After a user is registered, any of these values (that are stored in UDS characteristics) are accessible for modification.

Consent Code

The Consent Code is used to provide security for the user record. The Weight Scale doesn't grant access to the user record initially present in the example. To access the record, send the **Consent** command with Consent Code "0000". The Consent operation is also required when switching between the existing user records with the **SW2** button.

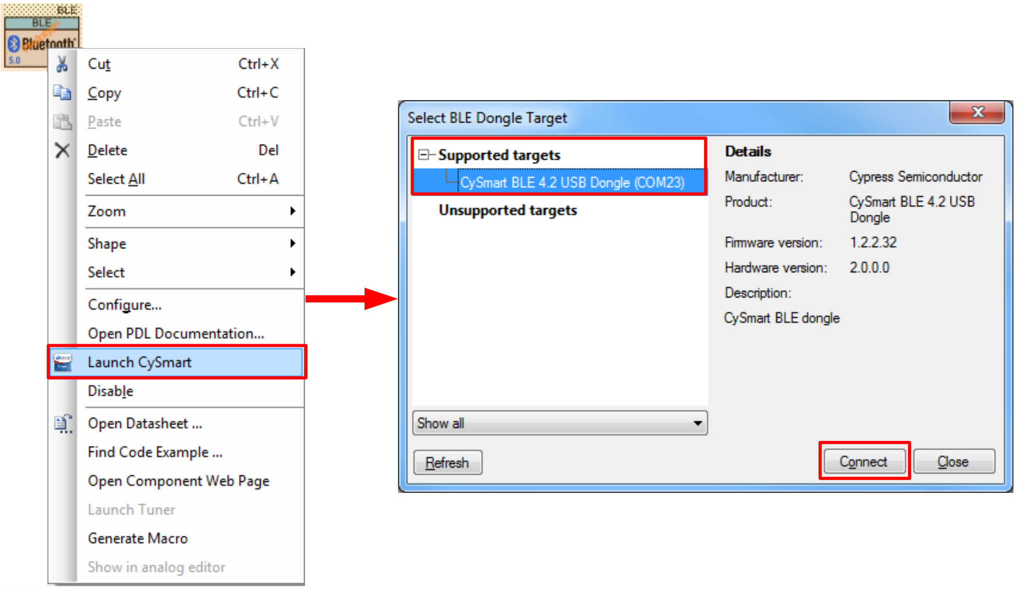
Refer to the Weight Scale Profile and Weight Scale Service specifications for more details.

You can use the CySmart app on a Windows PC, Android, or iOS BLE-compatible device as a Client for connection to the Weight Scale.

Operation Steps

1. Plug the CY8CKIT-062-BLE kit board into your computer's USB port.
2. Open a terminal window and perform following configuration: Baud rate – 115200, Parity – None, Stop bits – 1, Flow control – XON/XOFF. These settings must match the configuration of the PSoC Creator UART Component in the project.
3. Build the project and program it into the PSoC 6 MCU device. Choose **Debug** > **Program**. For more information on device programming, see PSoC Creator Help. Flash for both CPUs is programmed in a single program operation.
4. Observe the green LED blinks while the device is advertising, and the output in the terminal window.
5. Do the following to test example, using the CySmart Host Emulation Tool application as a Weight Scale Collector:
   a. Connect the BLE Dongle to your Windows PC. Wait for the driver installation to complete, if necessary.
   b. Launch the CySmart Host Emulation Tool by right-clicking on the BLE Component and selecting **Launch CySmart**. Alternatively, you can launch the tool by navigating to **Start** > **Programs** > **Cypress** and clicking on **CySmart**.
   c. CySmart automatically detects the BLE dongle connected to the PC. Click **Refresh** if the BLE dongle does not appear in the **Select BLE Dongle Target** pop-up window. Click **Connect**, as shown in Figure 1.

Figure 1. CySmart BLE Dongle Selection



CySmart BLE 4.2 USB Dongle (COM23)

Manufacturer: Cypress Semiconductor

Product: CySmart BLE 4.2 USB Dongle

Firmware version: 1.2.2.32

Hardware version: 2.0.0.0

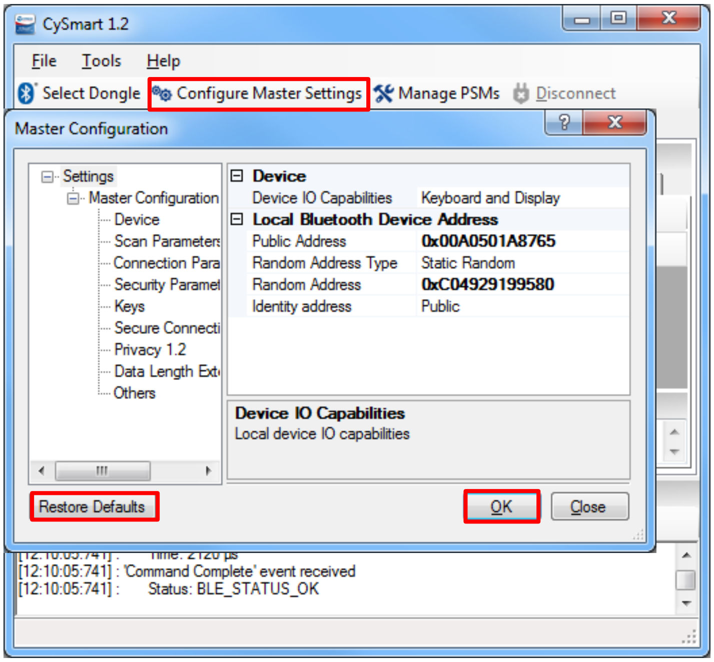
Description:
CySmart BLE dongle

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

-

d. Select **Configure Master Settings** and then click **Restore Defaults**, as Figure 2 shows. Then click **OK**.

Figure 2. CySmart Master Settings Configuration



Device	
Device IO Capabilities	Keyboard and Display
Local Bluetooth Device Address	
Public Address	**0x00A0501A8765**
Random Address Type	Static Random
Random Address	**0xC04929199580**
Identity address	Public

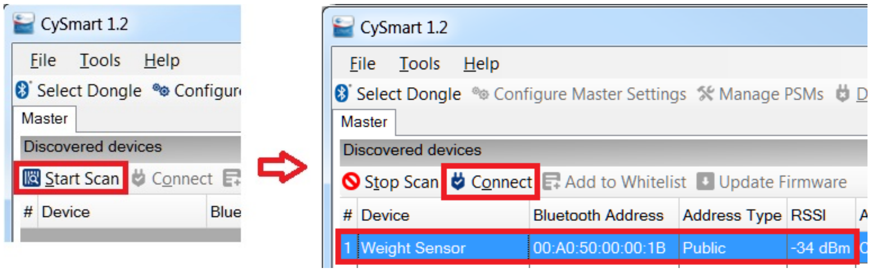
Device IO Capabilities
Local device IO capabilities

```
[12:10:05:741] :        Time: 2120 µs
[12:10:05:741] : 'Command Complete' event received
[12:10:05:741] :        Status: BLE_STATUS_OK
```

e. Press the reset switch on the Pioneer Kit to start BLE advertisement if no device is connected or device is in Hibernate mode (red LED is on). Otherwise, skip this step.

f. On the CySmart Host Emulation Tool, click **Start Scan**. Your device name (configured as **Weight Sensor**) should appear in the Discovered devices list, as Figure 3 shows. Select the device and click **Connect** to establish a BLE connection between the CySmart Host Emulation Tool and your device.

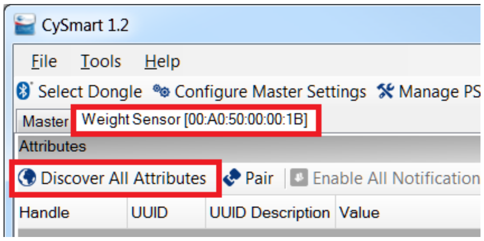
Figure 3. CySmart Device Discovery and Connection



#	Device	Bluetooth Address	Address Type	RSSI
1	Weight Sensor	00:A0:50:00:00:1B	Public	-34 dBm

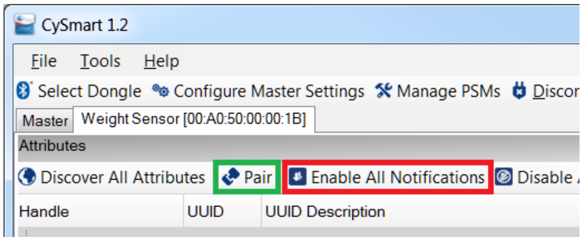
g. Once connected, switch to the '**Weight Sensor**' device tab and '**Discover all Attributes**' on your design from the CySmart Host Emulation Tool, as shown in Figure 4.

Figure 4. CySmart Attribute Discovery



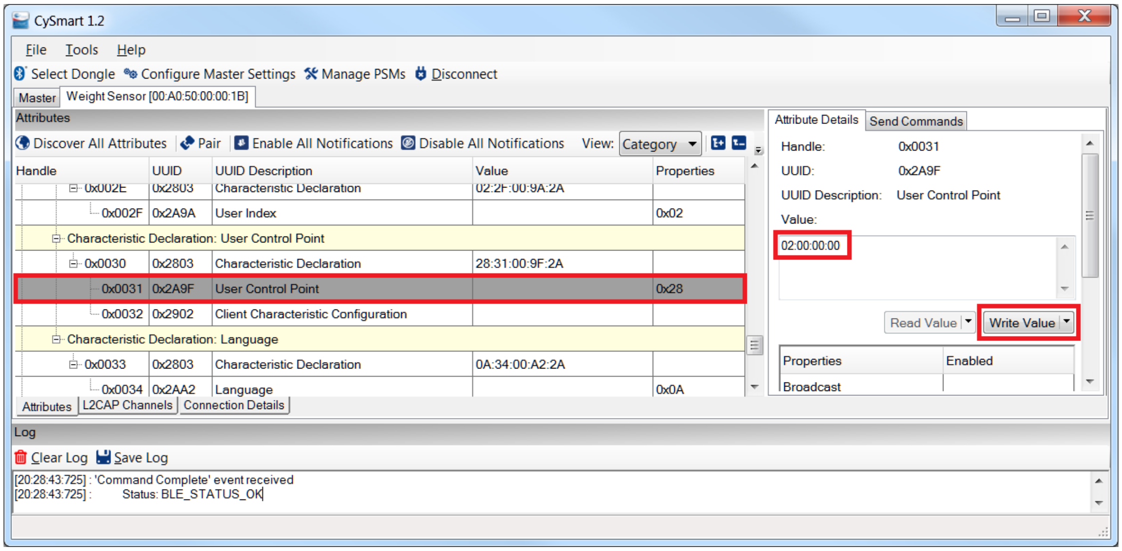
h. Click **Pair** after discovery finishes, then **Enable All Notifications** in the CySmart app as shown in Figure 5.

Figure 5. CySmart Pair and Enable All Notification



i. Locate the **User Control Point** Characteristic (UUID 0x2A9F) of the **User Data** Service and write the following value to the characteristic: **0x02000000.** The value represents the **Consent** command (0x02) with user index 0x00 and consent code 0x0000.

Figure 6. CySmart Windows App: User Control Point Indication – Consent Operation



Handle	UUID	UUID Description	Value	Properties
0x002E	0x2803	Characteristic Declaration	02:2F:00:9A:2A	
0x002F	0x2A9A	User Index		0x02
Characteristic Declaration: User Control Point				
0x0030	0x2803	Characteristic Declaration	28:31:00:9F:2A	
0x0031	0x2A9F	User Control Point		0x28
0x0032	0x2902	Client Characteristic Configuration		
Characteristic Declaration: Language				
0x0033	0x2803	Characteristic Declaration	0A:34:00:A2:2A	
0x0034	0x2AA2	Language		0x0A

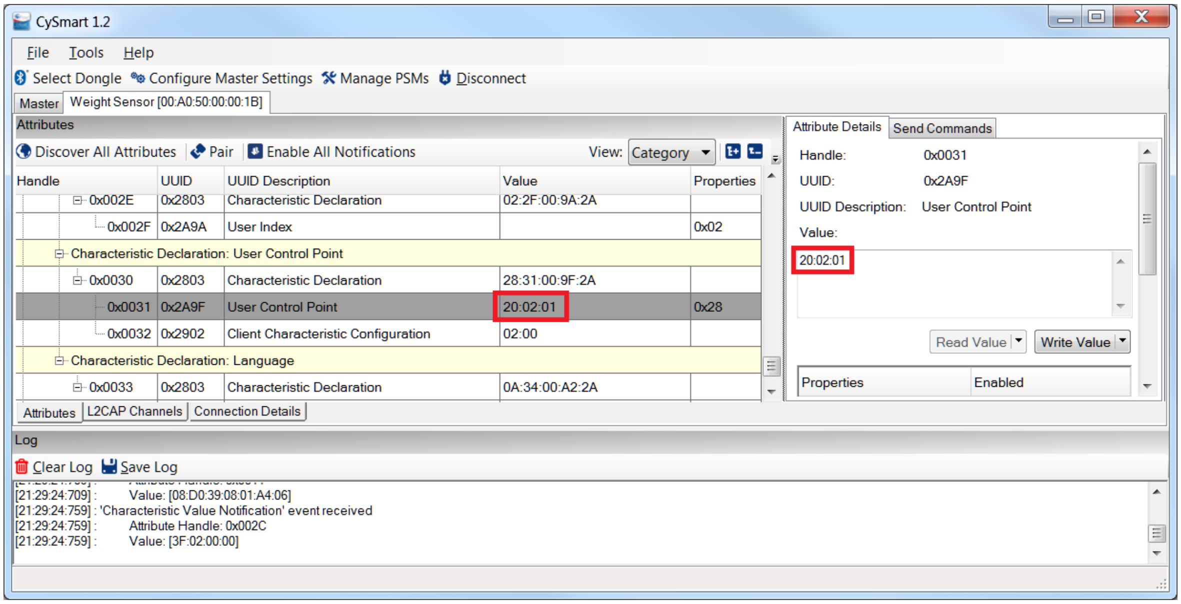
Handle: 0x0031
UUID: 0x2A9F
UUID Description: User Control Point
Value:
02:00:00:00

Properties	Enabled
Broadcast	

```
[20:28:43:725] : 'Command Complete' event received
[20:28:43:725] :        Status: BLE_STATUS_OK
```

j. Observe the response indication from the **User Control Point**. Figure 7. The general format of a response: XX:YY:ZZ [:PP], where XX – the response Op Code, YY- the requested Op Code, ZZ – the response value, PP – the response parameter (optional). The response value field can be set to one of the following values: 0x01 – Success, 0x02 – Op Code is not supported, 0x03 – Invalid Parameter, 0x04 – Operation Failed, 0x05 – User Not Authorized. Refer to the UDS specification for a detailed description.

Figure 7. CySmart Windows App: User Control Point Indication – Success Execution of Consent Operation



Handle	UUID	UUID Description	Value	Properties
0x002E	0x2803	Characteristic Declaration	02:2F:00:9A:2A	
0x002F	0x2A9A	User Index		0x02
Characteristic Declaration: User Control Point				
0x0030	0x2803	Characteristic Declaration	28:31:00:9F:2A	
0x0031	0x2A9F	User Control Point	20:02:01	0x28
0x0032	0x2902	Client Characteristic Configuration	02:00	
Characteristic Declaration: Language				
0x0033	0x2803	Characteristic Declaration	0A:34:00:A2:2A	

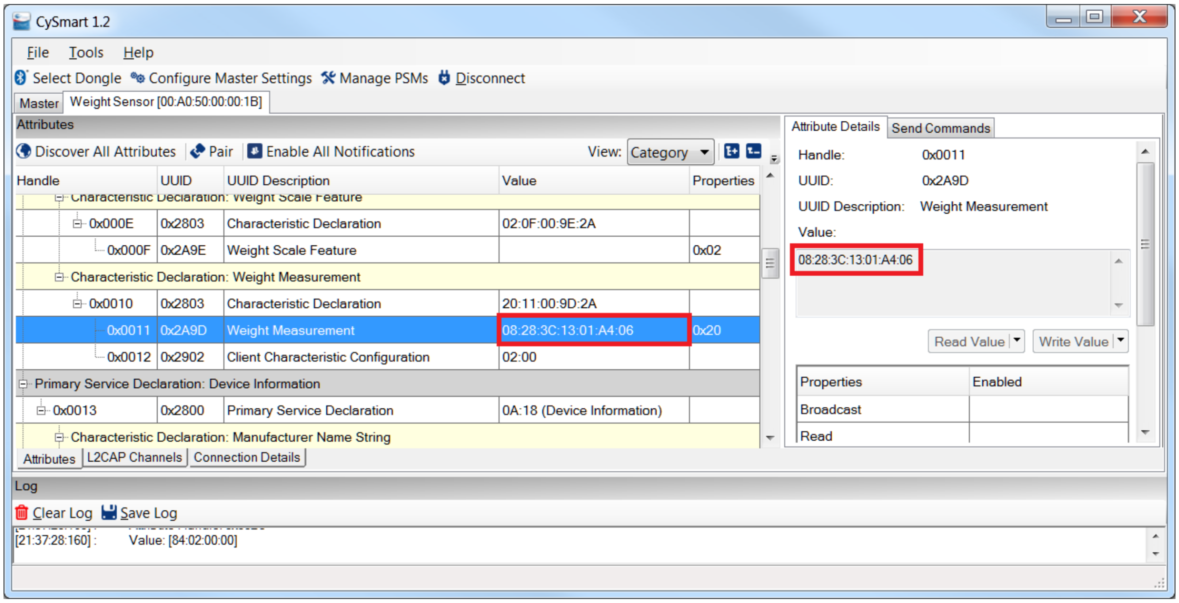
Handle: 0x0031
UUID: 0x2A9F
UUID Description: User Control Point
Value:
20:02:01

Properties	Enabled

```
[21:29:24:709] :        Value: [08:D0:39:08:01:A4:06]
[21:29:24:759] : 'Characteristic Value Notification' event received
[21:29:24:759] :        Attribute Handle: 0x002C
[21:29:24:759] :        Value: [3F:02:00:00]
```

k. Select the **Weight Measurement Characteristic** of the **Weight Scale Service** and observe notifications from the service.

Figure 8. CySmart Windows App: Weight Measurement Notification Received


CySmart 1.2
File Tools Help
Select Dongle Configure Master Settings Manage PSMs Disconnect
Master Weight Sensor [00:A0:50:00:00:1B]
Attributes
Discover All Attributes Pair Enable All Notifications View: Category
Attribute Details Send Commands
Read Value Write Value
Attributes L2CAP Channels Connection Details
Log
Clear Log Save Log

Handle	UUID	UUID Description	Value	Properties
Characteristic Declaration: Weight Measurement				
0x000E	0x2803	Characteristic Declaration	02:0F:00:9E:2A	
0x000F	0x2A9E	Weight Scale Feature		0x02
Characteristic Declaration: Weight Measurement				
0x0010	0x2803	Characteristic Declaration	20:11:00:9D:2A	
0x0011	0x2A9D	Weight Measurement	08:28:3C:13:01:A4:06	0x20
0x0012	0x2902	Client Characteristic Configuration	02:00	
Primary Service Declaration: Device Information				
0x0013	0x2800	Primary Service Declaration	0A:18 (Device Information)	
Characteristic Declaration: Manufacturer Name String				

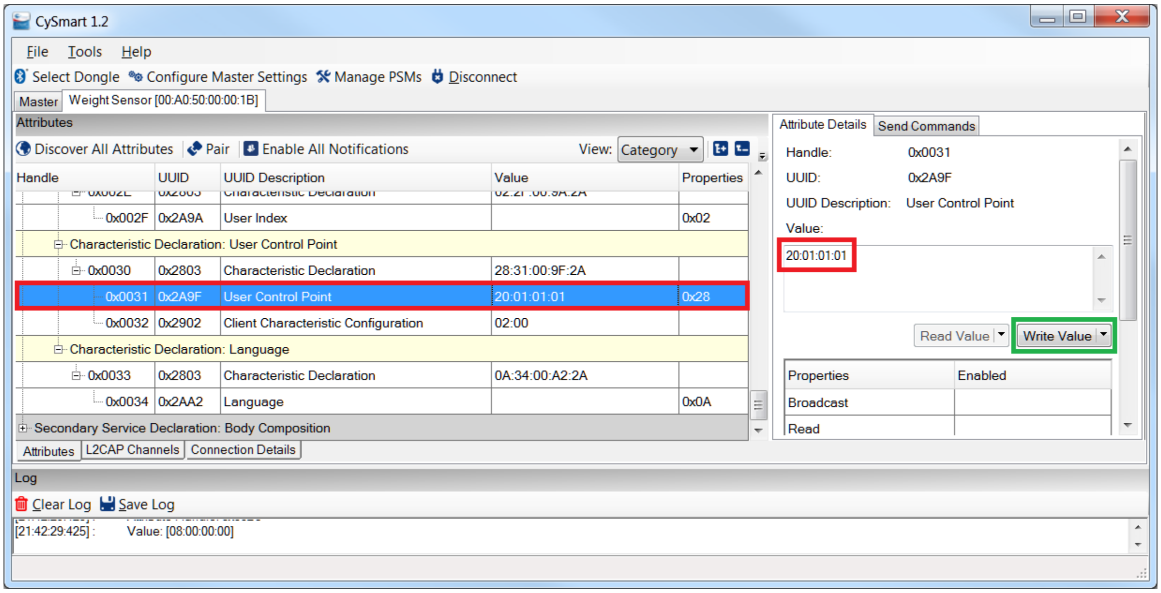
Handle: 0x0011
UUID: 0x2A9D
UUID Description: Weight Measurement
Value:
08:28:3C:13:01:A4:06

Properties	Enabled
Broadcast	
Read	

```
[21:37:28:160] :        Value: [84:02:00:00]
```

l. The **Consent** command can also be used to switch between the user records of the Weight Scale but the user should be registered: select **User Control Point** of the **User Data Service** and write the following value to the characteristic – **0x010000**. The value represents the Register New User command (0x01) with consent code of 0x0000. Refer to Consent Code**Error! Reference source not found.** section for more detail.

Figure 9. CySmart Windows App: User Control Point Indication – Successful Execution of Register New User Operation


CySmart 1.2
File Tools Help
Select Dongle Configure Master Settings Manage PSMs Disconnect
Master Weight Sensor [00:A0:50:00:00:1B]
Attributes
Discover All Attributes Pair Enable All Notifications View: Category
Attribute Details Send Commands
Read Value Write Value
Attributes L2CAP Channels Connection Details
Log
Clear Log Save Log

Handle	UUID	UUID Description	Value	Properties
0x002F	0x2A9A	User Index		0x02
Characteristic Declaration: User Control Point				
0x0030	0x2803	Characteristic Declaration	28:31:00:9F:2A	
0x0031	0x2A9F	User Control Point	20:01:01:01	0x28
0x0032	0x2902	Client Characteristic Configuration	02:00	
Characteristic Declaration: Language				
0x0033	0x2803	Characteristic Declaration	0A:34:00:A2:2A	
0x0034	0x2AA2	Language		0x0A
Secondary Service Declaration: Body Composition				

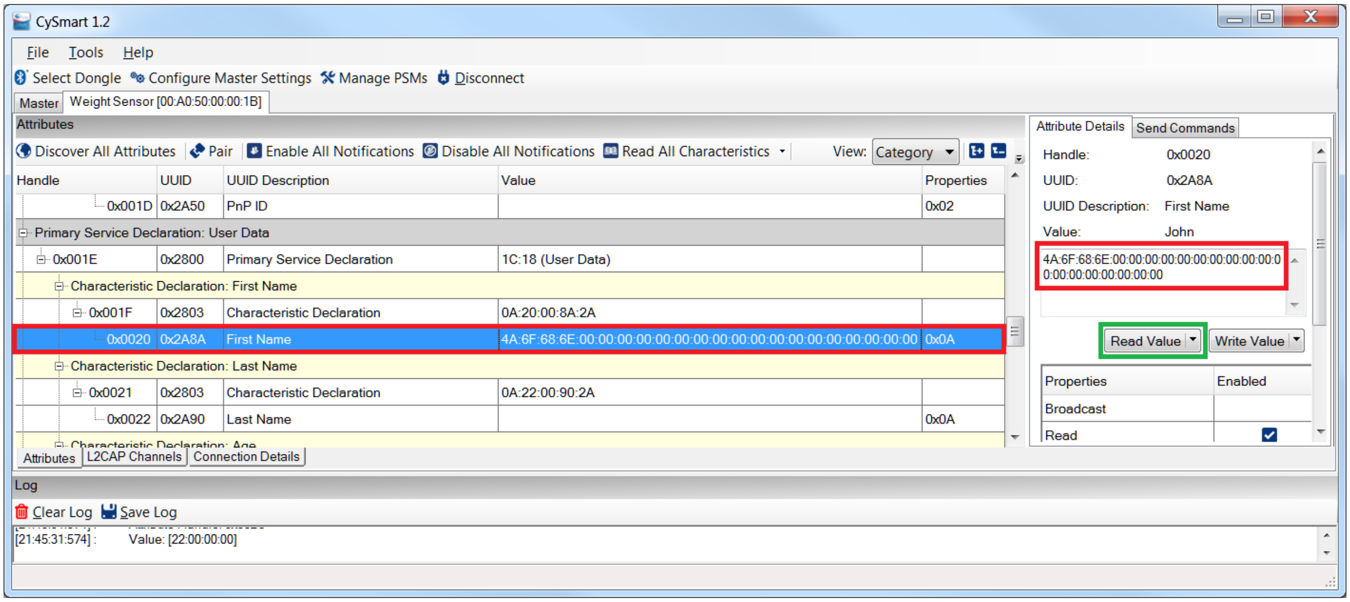
Handle: 0x0031
UUID: 0x2A9F
UUID Description: User Control Point
Value:
20:01:01:01

Properties	Enabled
Broadcast	
Read	

```
[21:42:29:425] :        Value: [08:00:00:00]
```

m. After indication of successful execution of the **Consent** or **Register New User** operation the UDS characteristics are accessible for read/write. Select the **First Name** Characteristic and click **Read Value**.

Figure 10. CySmart Windows App: Read UDS Characteristics – Read First Name



Handle	UUID	UUID Description	Value	Properties
0x001D	0x2A50	PnP ID		0x02
Primary Service Declaration: User Data				
0x001E	0x2800	Primary Service Declaration	1C:18 (User Data)	
Characteristic Declaration: First Name				
0x001F	0x2803	Characteristic Declaration	0A:20:00:8A:2A	
0x0020	0x2A8A	First Name	4A:6F:68:6E:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00	0x0A
Characteristic Declaration: Last Name				
0x0021	0x2803	Characteristic Declaration	0A:22:00:90:2A	
0x0022	0x2A90	Last Name		0x0A
Characteristic Declaration: Age				

Handle: 0x0020
UUID: 0x2A8A
UUID Description: First Name
Value: John

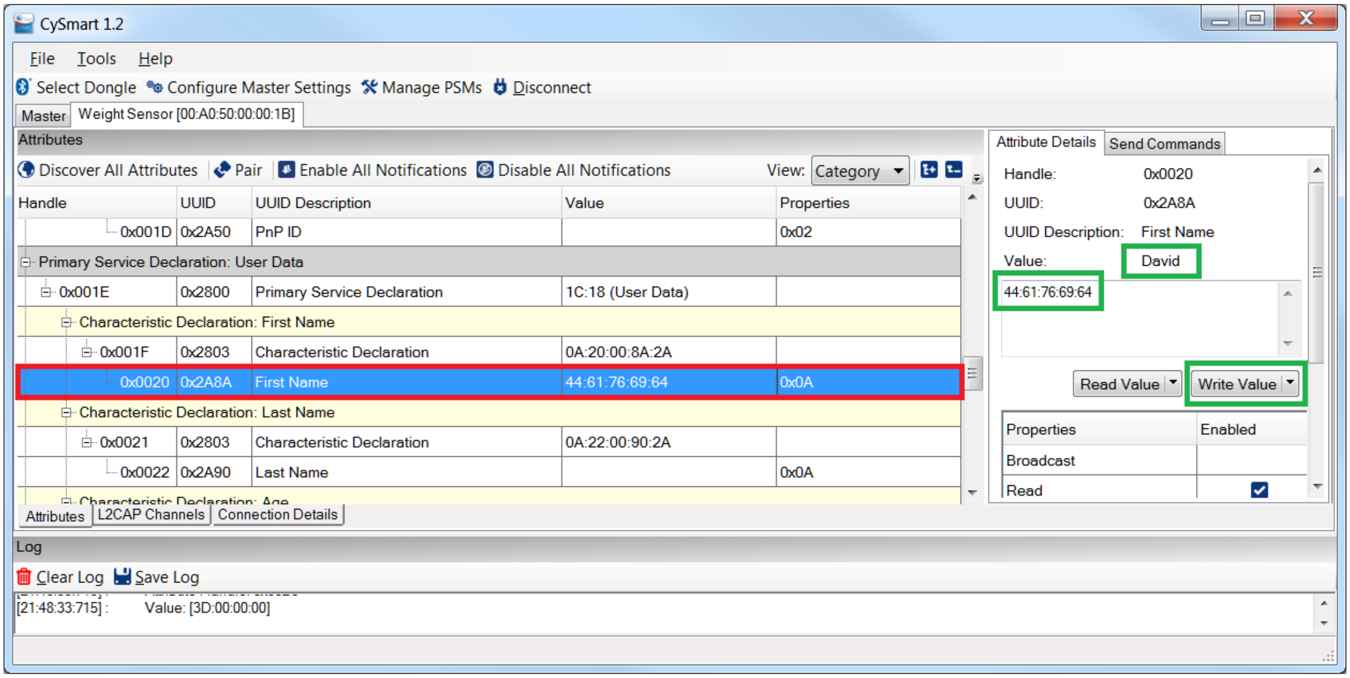
4A:6F:68:6E:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

Properties	Enabled
Broadcast	
Read	☑

```
[21:45:31:574] :        Value: [22:00:00:00]
```

n. Any of the UDS Characteristics can be written to modify the default values. To modify the **First Name** Characteristic, select it in the app, type the name converted to the ASCII format (e.g., “David” – 0x4461766964).

Figure 11. CySmart Windows App: Write UDS Characteristics – Read First Name



Handle	UUID	UUID Description	Value	Properties
0x001D	0x2A50	PnP ID		0x02
Primary Service Declaration: User Data				
0x001E	0x2800	Primary Service Declaration	1C:18 (User Data)	
Characteristic Declaration: First Name				
0x001F	0x2803	Characteristic Declaration	0A:20:00:8A:2A	
0x0020	0x2A8A	First Name	44:61:76:69:64	0x0A
Characteristic Declaration: Last Name				
0x0021	0x2803	Characteristic Declaration	0A:22:00:90:2A	
0x0022	0x2A90	Last Name		0x0A
Characteristic Declaration: Age				

Handle: 0x0020
UUID: 0x2A8A
UUID Description: First Name
Value: David

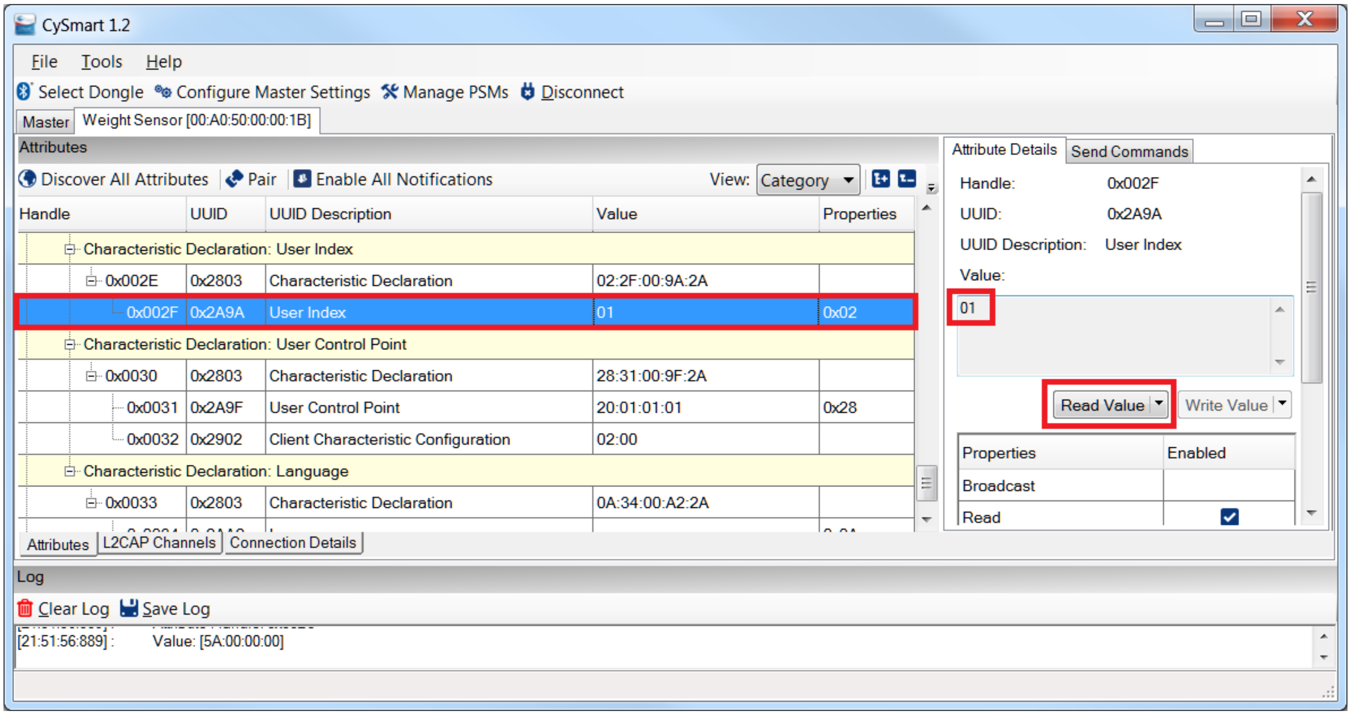
44:61:76:69:64

Properties	Enabled
Broadcast	
Read	☑

```
[21:48:33:715] :        Value: [3D:00:00:00]
```

o. Select **User Index Characteristic** and click **Read Value**. This will return the user index whose record is currently active in the Weight Scale.

Figure 12. CySmart Windows App: Read Database Change Increment and User Index Characteristics



Handle	UUID	UUID Description	Value	Properties
Characteristic Declaration: User Index				
0x002E	0x2803	Characteristic Declaration	02:2F:00:9A:2A	
0x002F	0x2A9A	User Index	01	0x02
Characteristic Declaration: User Control Point				
0x0030	0x2803	Characteristic Declaration	28:31:00:9F:2A	
0x0031	0x2A9F	User Control Point	20:01:01:01	0x28
0x0032	0x2902	Client Characteristic Configuration	02:00	
Characteristic Declaration: Language				
0x0033	0x2803	Characteristic Declaration	0A:34:00:A2:2A	

Handle: 0x002F
UUID: 0x2A9A
UUID Description: User Index
Value:
01

Properties	Enabled
Broadcast	
Read	☑

```
[21:51:56:889] :     Value: [5A:00:00:00]
```

p. Press the **SW2** button on the CY8CKIT-062 Pioneer Kit and read the value of **User Index Characteristic** again to see that the active user index was changed (at least two registered users are needed).

6. The CySmart mobile app (Android/iOS) does not have Weight Scale Profile implementation, but still can be used in GATT Data Base mode for test this example. You can repeat test flow for CySmart mobile app in step 5. Refer to Android and iOS CySmart User Guide.

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

   a. The code example ships with the UART debug port enabled. To disable it, set the macro DEBUG_UART_ ENABLED in *common.h* to DISABLED and rebuild the code.

   b. The output of the debug serial port looks like the sample below.

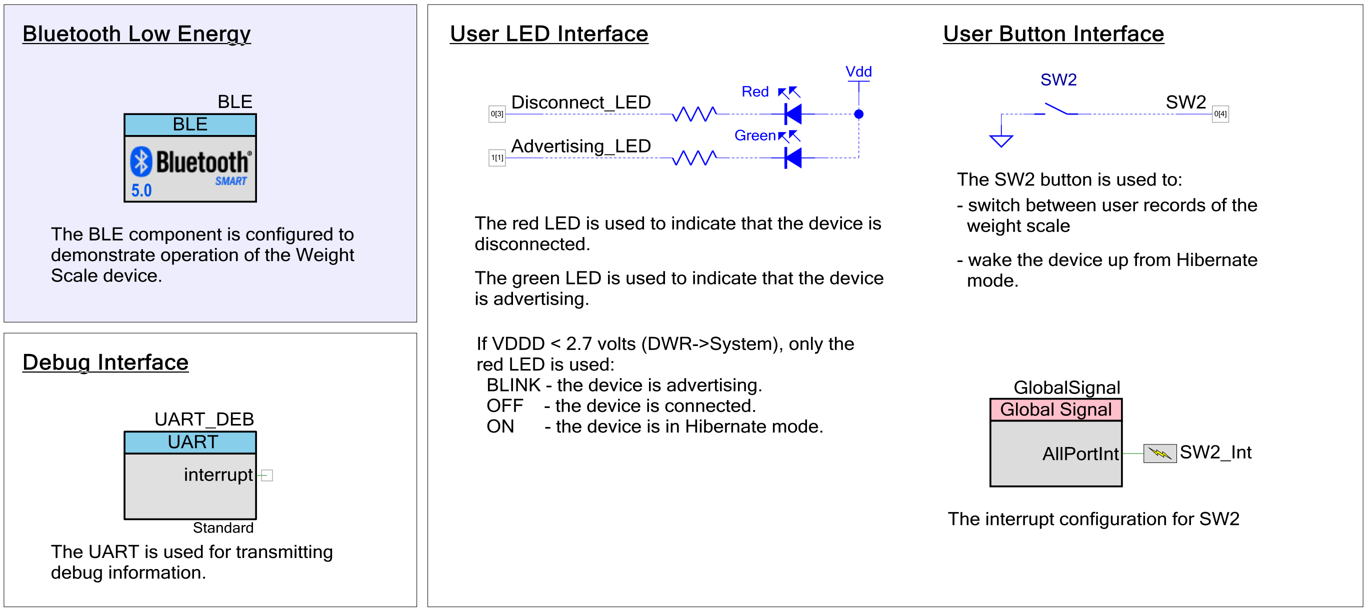
```
BLE Weight Scale code example
Body Composition Feature Characteristic was read successfully
Body Composition Measurement Characteristic was read successfully
Weight Feature Characteristic was read successfully
Weight Measurement Characteristic was read successfully
First Name Characteristic was read successfully
Last Name Characteristic was read successfully
Age Characteristic was read successfully
Gender Characteristic was read successfully
Weight Characteristic was read successfully
Height Characteristic was read successfully
Database Change Increment Characteristic was read successfully

BLE Stack Version: 5.0.1.899
CY_BLE_EVT_STACK_ON
CY_BLE_EVT_SET_DEVICE_ADDR_COMPLETE
CY_BLE_EVT_LE_SET_EVENT_MASK_COMPLETE
CY_BLE_EVT_GET_DEVICE_ADDR_COMPLETE: public:00a05000001b
CY_BLE_EVT_SET_TX_PWR_COMPLETE
CY_BLE_EVT_SET_TX_PWR_COMPLETE
CY_BLE_EVT_GAPP_ADVERTISEMENT_START_STOP, status: 0, state: 2
CY_BLE_EVT_GAP_KEYS_GEN_COMPLETE
```

```
CY_BLE_EVT_GAPP_UPDATE_ADV_SCAN_DATA_COMPLETE, status: 0
CY_BLE_EVT_GATT_CONNECT_IND: 3, 7
CY_BLE_EVT_GAP_DEVICE_CONNECTED: 0, 7( ms), 0, a
CY_BLE_EVT_GATTS_XCNHG_MTU_REQ, final mtu= 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ, attHandle: 3
CY_BLE_EVT_GAP_AUTH_REQ: security=0x1, bonding=0x1, ekeySize=0x10, err=0x0
CY_BLE_EVT_GAP_SMP_NEGOTIATED_AUTH_INFO: security:1, bonding:1, ekeySize:10, authErr 0
ENCRYPT_CHANGE: 1
CY_BLE_EVT_GAP_KEYINFO_EXCHNGE_CMPLT
CY_BLE_EVT_GAP_AUTH_COMPLETE: security: 0x1, bonding: 0x1, ekeySize: 0x10, authErr 0x0
CY_BLE_EVT_PENDING_FLASH_WRITE
Store bonding data, status: 0, pending: 0

New measurements from weight sensor
New weight: 73.00 kg BMI: 26. 0 %
Indication wasn't sent. Indications are disabled.

CY_BLE_EVT_GATTS_INDICATION_ENABLED
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ, attHandle: 11
CY_BLE_EVT_WSSS_INDICATION_ENABLED
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ, attHandle: 18
CY_BLE_EVT_UDSS_INDICATION_ENABLED
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ, attHandle: 50
CY_BLE_EVT_WSSS_INDICATION_CONFIRMED
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ, attHandle: 45
CY_BLE_EVT_BCSS_INDICATION_ENABLED
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ, attHandle: 58
CY_BLE_EVT_BCSS_INDICATION_ENABLED
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ, attHandle: 58
CY_BLE_EVT_BCSS_INDICATION_ENABLED
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ, attHandle: 58

CY_BLE_EVT_UDSS_WRITE_CHAR
Char index: 29
Access allowed for: John, Smith (Index - 0).

New measurements from weight sensor
New weight: 78.50 kg BMI: 28. 0 %
CY_BLE_EVT_WSSS_INDICATION_CONFIRMED
New measurements from weight sensor
New weight: 79.00 kg BMI: 28. 2 %
CY_BLE_EVT_WSSS_INDICATION_CONFIRMED
New measurements from weight sensor
New weight: 79.50 kg BMI: 28. 3 %
CY_BLE_EVT_WSSS_INDICATION_CONFIRMED
New measurements from weight sensor
New weight: 80.00 kg BMI: 28. 5 %
```

Design and Implementation

The project demonstrates the core functionality of the BLE Component configured as the Weight Scale in the GAP Peripheral role. The Weight Scale Sensor uses one instance of the Weight Scale Service (WSS), Body Composition Service (BCS), User Data Service (UDS), and Device Information Service (DIS) to simulate weight measurements for up to four registered users. The Weight Scale Sensor operates with other devices that implement the Weight Scale Collector Profile. Figure 13 shows the top design schematic.

Figure 13. BLE Weight-Scale Profile Example Schematic



The BLE component is configured to demonstrate operation of the Weight Scale device.

The UART is used for transmitting debug information.

The red LED is used to indicate that the device is disconnected.

The green LED is used to indicate that the device is advertising.

If VDDD < 2.7 volts (DWR->System), only the red LED is used:
BLINK - the device is advertising.
OFF - the device is connected.
ON - the device is in Hibernate mode.

The SW2 button is used to:

- switch between user records of the weight scale
- wake the device up from Hibernate mode.

The interrupt configuration for SW2

After a startup, the device performs BLE Component initialization. In this project, three callback functions are required for BLE operation. The `AppCallBack()` callback function is required to receive generic events from the BLE stack and the service-specific callbacks `WssCallBack()`, `BcsCallBack()`, and `UdsCallBack()` are required for WSS, BCS, and UDS service-specific events accordingly. The `CY_BLE_EVT_STACK_ON` event indicates a successful initialization of the BLE stack. After this event is received, the Component starts advertising with the packet structure as configured in BLE Component Customizer.

After the 30-second advertising period expires, the Component switches to slow advertisement parameters. The BLE Component stops advertising after the 180-second advertising period expires.

On an advertisement timeout, the system remains in Hibernate mode. Press **SW2** to wake the system and start re-advertising.

To delete the pair information from **Bond List** press and hold **SW2** for 4 seconds

The Weight Scale device can be connected to any BLE (4.0 or later)-compatible device configured as the GAP Central role and GATT Client which supports the Weight Scale Profile. The Device Information services may be optionally used.

To connect to the Weight Scale device, send a connection request to the device while the device is advertising. The green LED blinks while the device is advertising. The red LED is turned ON after disconnection to indicate that no Client is connected to the device. When the Client connects successfully, the red and green LEDs are turned OFF.

While connected to the Client and between the connection intervals, the device is put into Deep Sleep mode.

Pin assignments

Pin assignments and connections required on the development board for supported kits are in Table 1.

Table 1. Pin Assignment

Pin Name	Development Kit	Comment
	CY8CKIT-062	
\UART_DEB:rx\	P5[0]	
\UART_DEB:tx\	P5[1]	
\UART_DEB:rts\	P5[2]	
\UART_DEB:cts\	P5[3]	
Disconnect_LED	P0[3]	The red color of the RGB LED
Advertising_LED	P1[1]	The green color of the RGB LED
SW2	P0[4]	

Components and Settings

Table 2 lists the PSoC Creator Components used in this example, how they are used in the design, and the non-default settings required so they function as intended.

Table 2. PSoC Creator Components

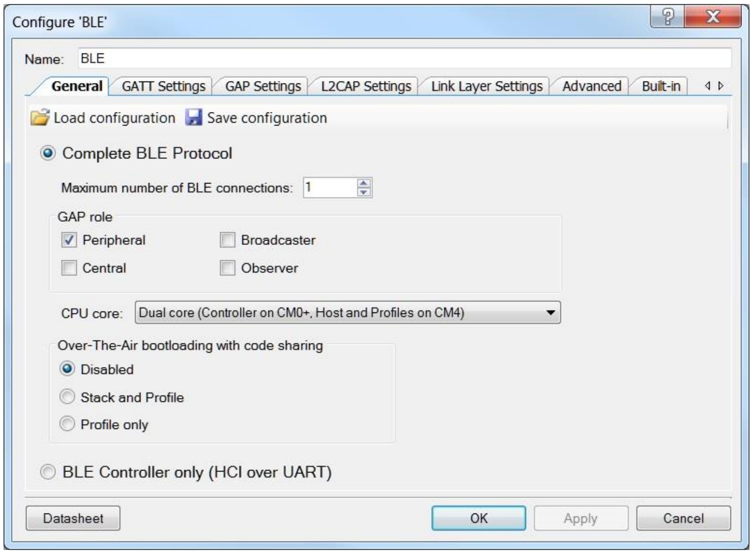
Component	Instance Name	Purpose	Non-default Settings
Bluetooth Low Energy (BLE)	BLE	The BLE component is configured to demonstrate operation of the Weight Scale device.	Refer to Parameter Settings section
Digital Input Pin	SW2	This pin is used to generate interrupts when the user button (**SW2**) is pressed.	**[General tab]** Uncheck HW connection Drive mode: Resistive Pull Up
Digital Output pin	Disconnect_LED Advertising_LED	These GPIOs are configured as firmware-controlled digital output pins that control LEDs.	**[General tab]** Uncheck HW connection Drive mode: Strong Drive
SysInt	SW2_Int	This Component is configured to extract interrupts from GlobalSignal.	Default
GSRef	GlobalSignal	This Component is used to detect if any of the interrupt enabled pins triggered an interrupt. It is a separate resource from the dedicated port interrupts, and it has the ability to wake up the chip from deep-sleep mode	**[General tab]** Global signal name: HWCombined Port Interrupt (AllPortInt)
UART (SCB)	UART_DEBUG	This Component is used to print messages on a terminal program.	Default

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

Parameter Settings

The BLE Component is configured as the Weight Scale (GATT Server) in the GAP Peripheral role. In the following figures, settings that differ from default values are highlighted in red.

Figure 14. General Settings



Configure 'BLE'

Name: BLE

● Complete BLE Protocol

Maximum number of BLE connections: 1

GAP role

☑ Peripheral □ Broadcaster

□ Central □ Observer

CPU core: Dual core (Controller on CM0+, Host and Profiles on CM4)

Over-The-Air bootloading with code sharing

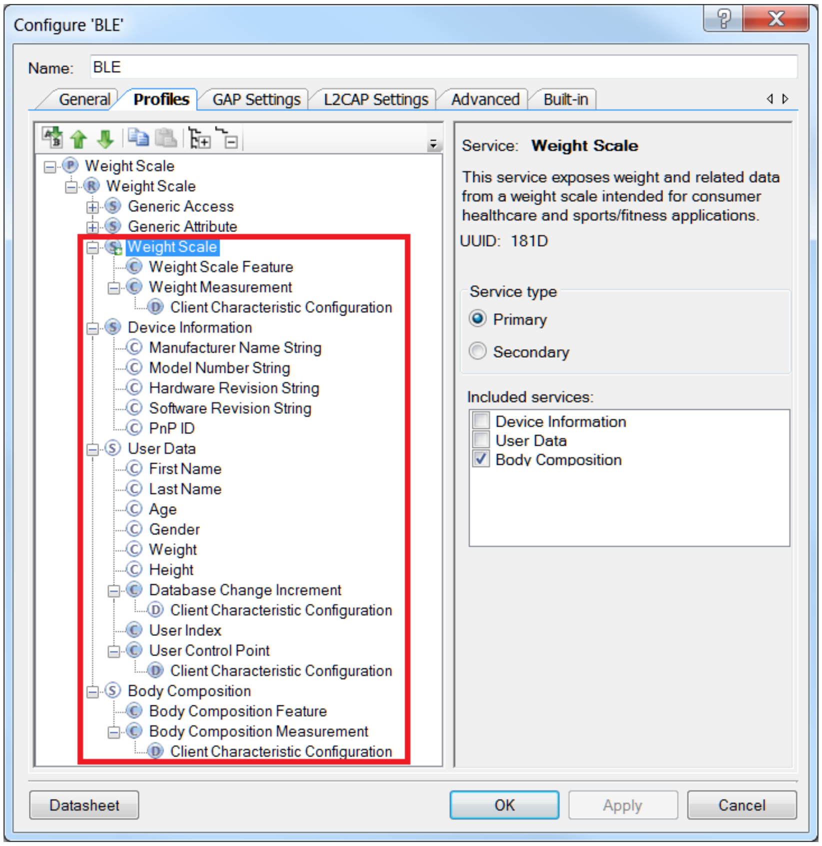
● Disabled

○ Stack and Profile

○ Profile only

○ BLE Controller only (HCI over UART)

Figure 15. GATT Settings



Configure 'BLE'

Name: BLE

- Weight Scale
  - Weight Scale
    - Generic Access
    - Generic Attribute
    - Weight Scale
      - Weight Scale Feature
      - Weight Measurement
        - Client Characteristic Configuration
    - Device Information
      - Manufacturer Name String
      - Model Number String
      - Hardware Revision String
      - Software Revision String
      - PnP ID
    - User Data
      - First Name
      - Last Name
      - Age
      - Gender
      - Weight
      - Height
      - Database Change Increment
        - Client Characteristic Configuration
      - User Index
      - User Control Point
        - Client Characteristic Configuration
    - Body Composition
      - Body Composition Feature
      - Body Composition Measurement
        - Client Characteristic Configuration

Service: **Weight Scale**

This service exposes weight and related data from a weight scale intended for consumer healthcare and sports/fitness applications.

UUID: 181D

Service type

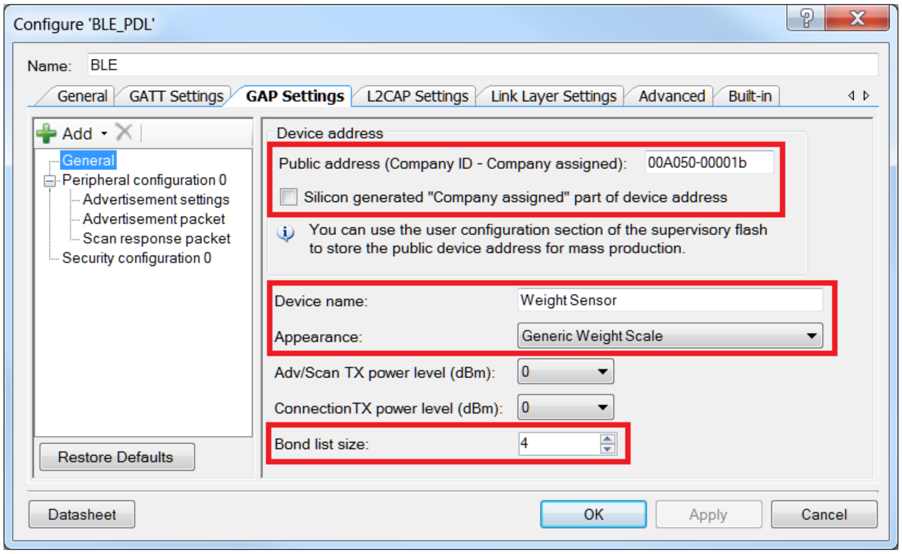
● Primary

○ Secondary

Included services:

□ Device Information
□ User Data
☑ Body Composition

Figure 16. GAP Settings



Configure 'BLE_PDL'

Name: BLE

Device address

Public address (Company ID - Company assigned): 00A050-00001b

□ Silicon generated "Company assigned" part of device address

You can use the user configuration section of the supervisory flash to store the public device address for mass production.

Device name: Weight Sensor

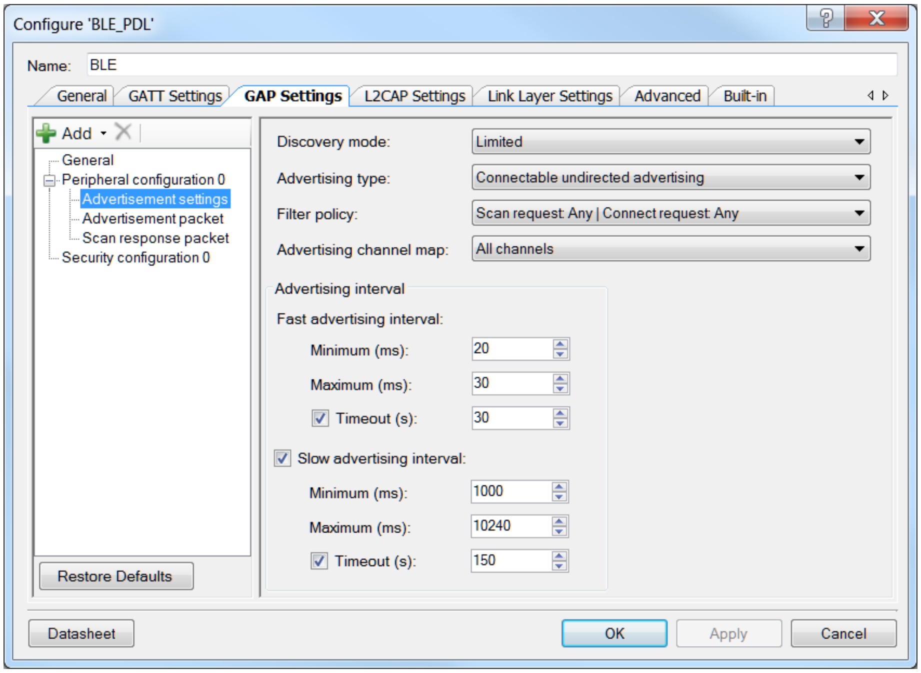
Appearance: Generic Weight Scale

Adv/Scan TX power level (dBm): 0

ConnectionTX power level (dBm): 0

Bond list size: 4

Figure 17. Advertisement Settings



Configure 'BLE_PDL'

Name: BLE

Discovery mode: Limited

Advertising type: Connectable undirected advertising

Filter policy: Scan request: Any | Connect request: Any

Advertising channel map: All channels

Advertising interval

Fast advertising interval:

Minimum (ms): 20

Maximum (ms): 30

☑ Timeout (s): 30

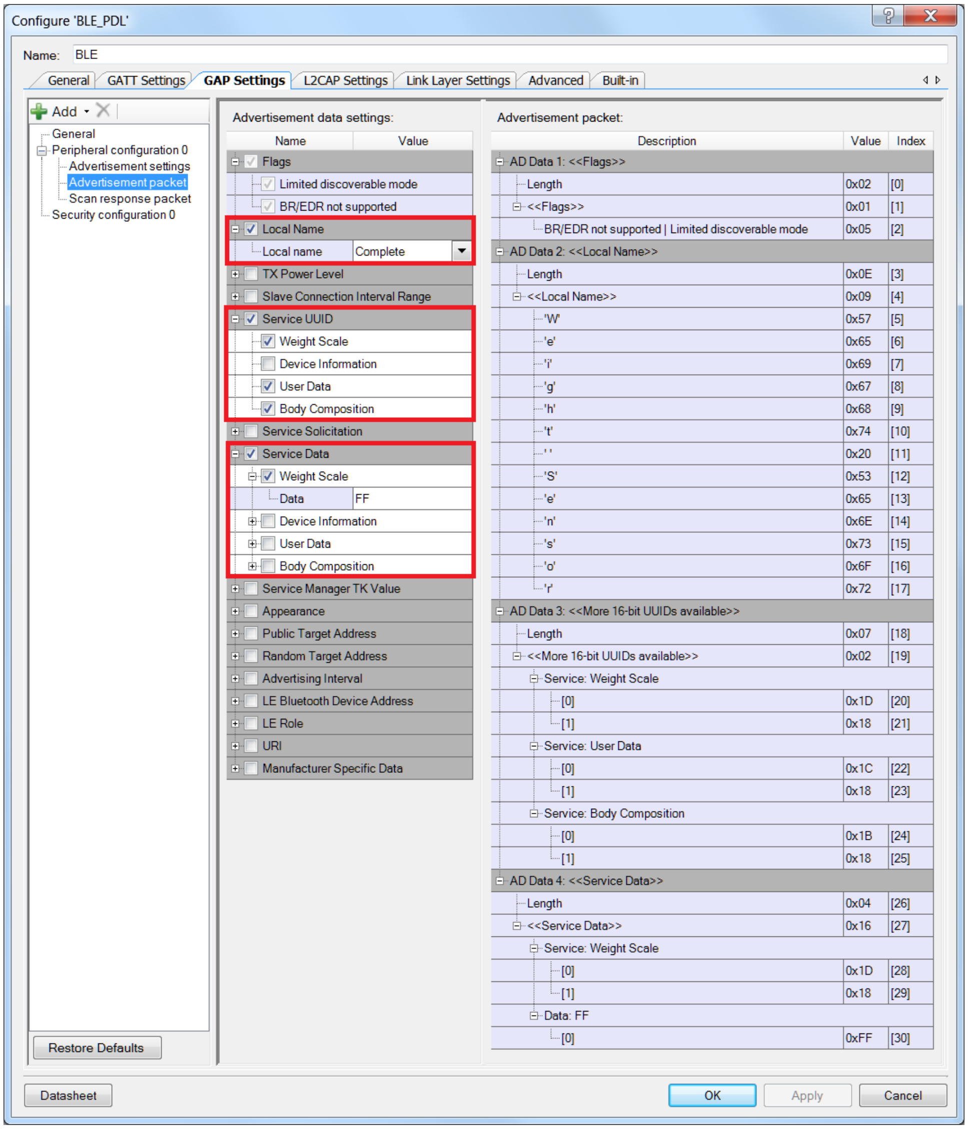
☑ Slow advertising interval:

Minimum (ms): 1000

Maximum (ms): 10240

☑ Timeout (s): 150

Figure 18. Advertisement Packet



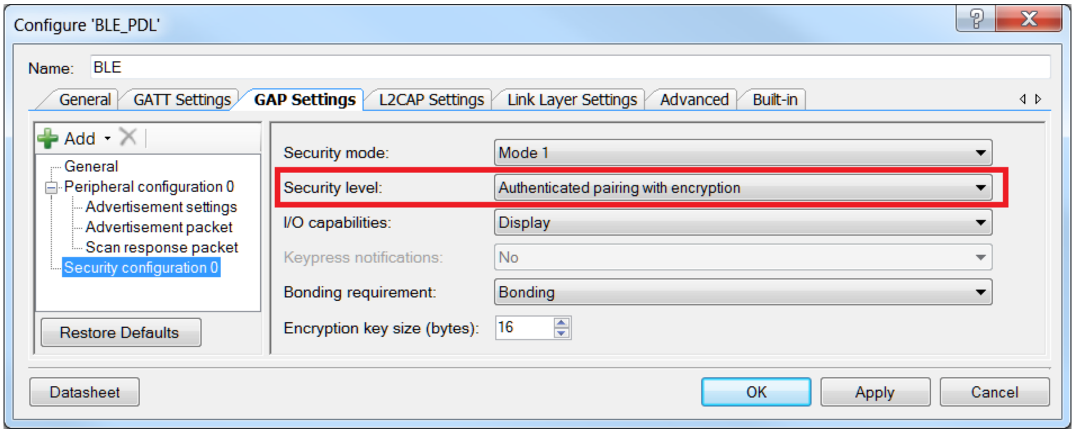
Advertisement data settings:

Name	Value
☑ Flags	
☑ Limited discoverable mode	
☑ BR/EDR not supported	
☑ Local Name	
Local name	Complete
□ TX Power Level	
□ Slave Connection Interval Range	
☑ Service UUID	
☑ Weight Scale	
□ Device Information	
☑ User Data	
☑ Body Composition	
□ Service Solicitation	
☑ Service Data	
☑ Weight Scale	
Data	FF
□ Device Information	
□ User Data	
□ Body Composition	
□ Service Manager TK Value	
□ Appearance	
□ Public Target Address	
□ Random Target Address	
□ Advertising Interval	
□ LE Bluetooth Device Address	
□ LE Role	
□ URI	
□ Manufacturer Specific Data	

Advertisement packet:

Description	Value	Index
AD Data 1: <<Flags>>		
Length	0x02	[0]
<<Flags>>	0x01	[1]
BR/EDR not supported | Limited discoverable mode	0x05	[2]
AD Data 2: <<Local Name>>		
Length	0x0E	[3]
<<Local Name>>	0x09	[4]
'W'	0x57	[5]
'e'	0x65	[6]
'i'	0x69	[7]
'g'	0x67	[8]
'h'	0x68	[9]
't'	0x74	[10]
' '	0x20	[11]
'S'	0x53	[12]
'e'	0x65	[13]
'n'	0x6E	[14]
's'	0x73	[15]
'o'	0x6F	[16]
'r'	0x72	[17]
AD Data 3: <<More 16-bit UUIDs available>>		
Length	0x07	[18]
<<More 16-bit UUIDs available>>	0x02	[19]
Service: Weight Scale		
[0]	0x1D	[20]
[1]	0x18	[21]
Service: User Data		
[0]	0x1C	[22]
[1]	0x18	[23]
Service: Body Composition		
[0]	0x1B	[24]
[1]	0x18	[25]
AD Data 4: <<Service Data>>		
Length	0x04	[26]
<<Service Data>>	0x16	[27]
Service: Weight Scale		
[0]	0x1D	[28]
[1]	0x18	[29]
Data: FF		
[0]	0xFF	[30]

Figure 19. Security Settings



Security mode:	Mode 1
Security level:	Authenticated pairing with encryption
I/O capabilities:	Display
Keypress notifications:	No
Bonding requirement:	Bonding
Encryption key size (bytes):	16

Switching the CPU Cores Usage

This section describes how to switch between different CPU cores usage (Single core/ Dual core) in the BLE PDL examples.

The BLE component has the CPU Core parameter that defines the cores usage. It can take the following values:

- Single core (Complete Component on CM0+) – only CM0+ core will be used.
- Single core (Complete Component on CM4) – only CM4 core will be used.
- Dual core (Controller on CM0+, Host and Profiles on CM4) – both cores will be used: CM0+ for the Controller and CM4 for the Host and Profiles.

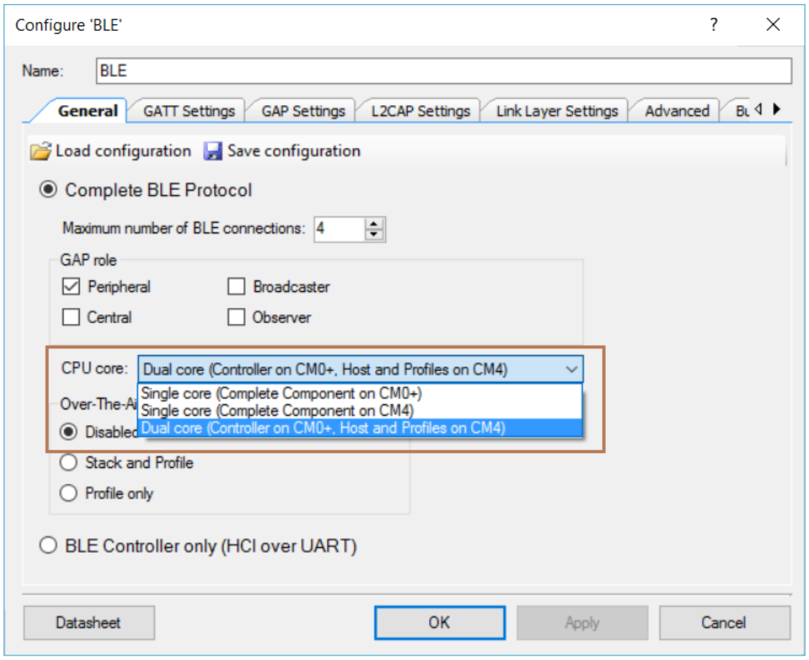
The BLE example structure allows easy switching between different CPU cores options. Important to remember:

- All application host-files must be run on the host core.
- The BLESS interrupt must be assigned to the core where the controller runs.
- All additional interrupts (SW2, MCWDT, etc.) used in the example must be assigned to the host core.

Steps for switching the CPU Cores usage:

1. In the BLE customizer **General** tab, select appropriate CPU core option.

Figure 20. Select CPU Core



Configure 'BLE'

Name: BLE

● Complete BLE Protocol

Maximum number of BLE connections: 4

GAP role

☑ Peripheral □ Broadcaster

□ Central □ Observer

CPU core: Dual core (Controller on CM0+, Host and Profiles on CM4)

Single core (Complete Component on CM0+)
Single core (Complete Component on CM4)
Dual core (Controller on CM0+, Host and Profiles on CM4)

Over-The-Ai

● Disabled

○ Stack and Profile

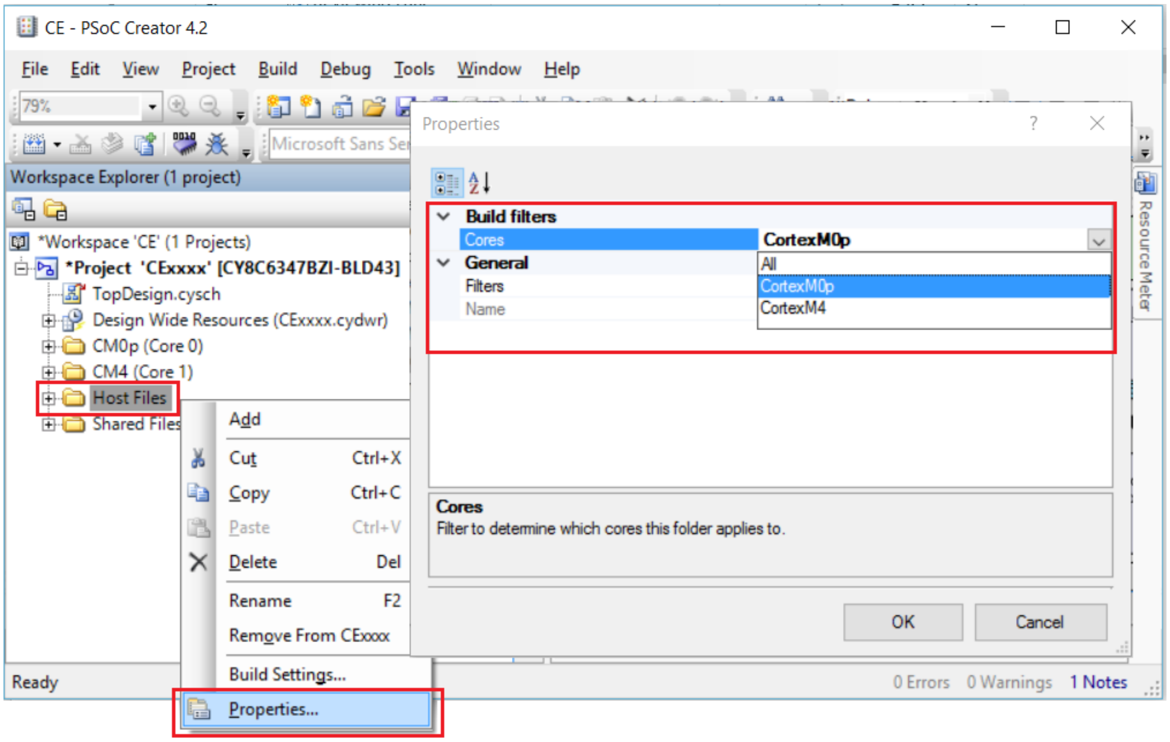
○ Profile only

○ BLE Controller only (HCI over UART)

2. Identify the core on which host files will run. In the workspace explorer panel, right click **Host Files**, choose **Properties**. Set the **Cores** property corresponding to the CPU core chosen in step 1, as shown in Figure 21.

   - for Single core (Complete Component on CM0+) option – CM0+
   - for Single core (Complete Component on CM4) option – CM4
   - for Dual core (Controller on CM0+, Host and Profiles on CM4) option – CM4

Figure 21. Change Core Properties



Build filters

Cores	CortexM0p

General

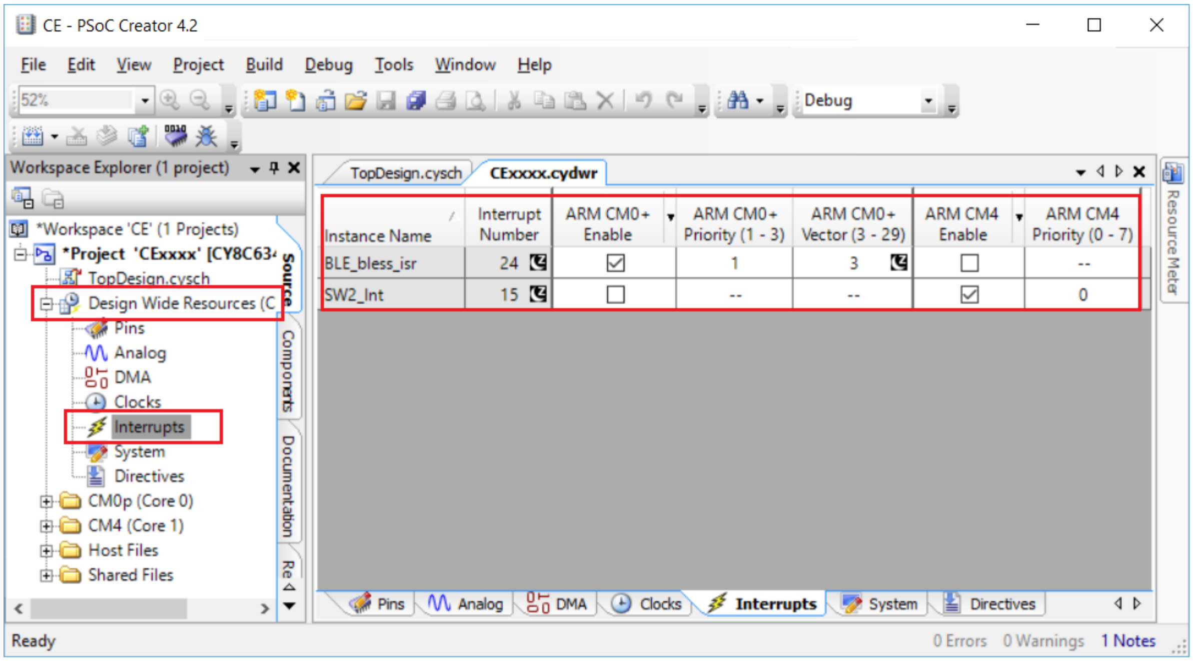
Filters	
Name	

Cores
Filter to determine which cores this folder applies to.

3. Assign the BLE_bless_isr and other peripheral (button – SW2, timer(s) etc.) interrupts to appropriate core in DWR-> interrupts tab:

- for **Single core (Complete Component on CM0+)** option: BLE_bless_isr and peripheral interrupts on **CM0+**
- for **Single core (Complete Component on CM4)** option: BLE_bless_isr and peripheral interrupts on **CM4**
- for **Dual core (Controller on CM0+, Host and Profiles on CM4)** option: BLE_bless_isr interrupt on **CM0+,** other peripheral interrupts on **CM4**

Figure 22. Assign Interrupts



Instance Name	Interrupt Number	ARM CM0+ Enable	ARM CM0+ Priority (1 - 3)	ARM CM0+ Vector (3 - 29)	ARM CM4 Enable	ARM CM4 Priority (0 - 7)
BLE_bless_isr	24	☑	1	3	☐	--
SW2_Int	15	☐	--	--	☑	0

Reusing This Example

This example is designed for the CY8CKIT-062-BLE pioneer kit. To port the design to a different PSoC 6 MCU device and/or kit, change the target device using the Device Selector and update the pin assignments in the Design Wide Resources Pins settings as needed.

Related Documents

The following table lists all relevant application notes, code examples, knowledge base articles, device datasheets and Component datasheets.

Table 3. Related Documents

Application Notes		
AN210781	Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 6 BLE, and how to build a basic code example.
AN215656	PSoC 6 MCU Dual- CPU System Design	Presents the theory and design considerations related to this code example.
Software and Drivers		
CySmart – Bluetooth® LE Test and Debug Tool		CySmart is a Bluetooth® LE host emulation tool for Windows PCs. The tool provides an easy-to-use Graphical User Interface (GUI) to enable the user to test and debug their Bluetooth LE peripheral applications.
PSoC Creator Component Datasheets		
Bluetooth Low Energy (BLE_PDL) Component		The Bluetooth Low Energy (BLE_PDL) Component provides a comprehensive GUI-based configuration window to facilitate designing applications requiring BLE connectivity.
Device Documentation		
PSoC® 6 MCU: PSoC 63 with BLE. Datasheet.		PSoC® 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
Development Kit (DVK) Documentation		
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit		

Document History

Document Title: CE217645 - BLE Weight Scale Profile with PSoC 6 MCU with BLE Connectivity

Document Number: 002-17645

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6092395	NPAL	03/15/2018	New spec

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 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

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 does not assume any liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. 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. 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.