CE215120 - BLE Device Information Service with PSoC 6 MCU with BLE Connectivity

Objective

This example project demonstrates how to configure and use the Bluetooth Low Energy (BLE) Component APIs and an application layer callback.

Overview

This is a simple BLE example project that demonstrates how to configure and use the BLE Component APIs and an application layer callback. The Device Information Service is used as an example to demonstrate how to configure the BLE service characteristics in the BLE Component.

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. See 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 V_{DDD} Voltage < 2.7 V

If the V_{DDD} voltage is set to lesser than 2.7 V 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

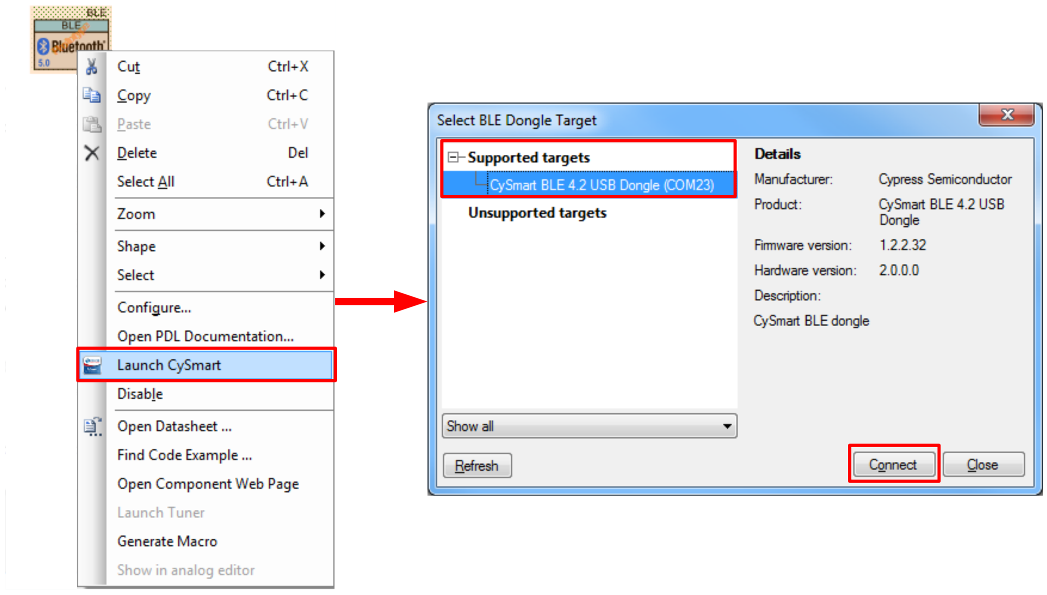
Operation

The green LED blinks to indicate that the device is advertising. The red LED turns ON to indicate that no client is connected to the device. When a client is connected successfully, the red and green LEDs are turned OFF.

Operation Steps

1. Plug the CY8CKIT-062-BLE kit board into your computer's USB port.
2. 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.
3. Observe the green LED blinks while the device is advertising.
4. Do the following to the test example using the CySmart Host Emulation Tool application as a Device Information Service Client:
   a. Connect the BLE Dongle to your Windows PC. Wait for the driver installation to complete, if necessary.
   b. Right-click the BLE Component and select **Launch CySmart** to launch the CySmart Host Emulation Tool. Alternatively, you can navigate to **Start** > **Programs** > **Cypress** and click **CySmart** to launch the tool.
   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.

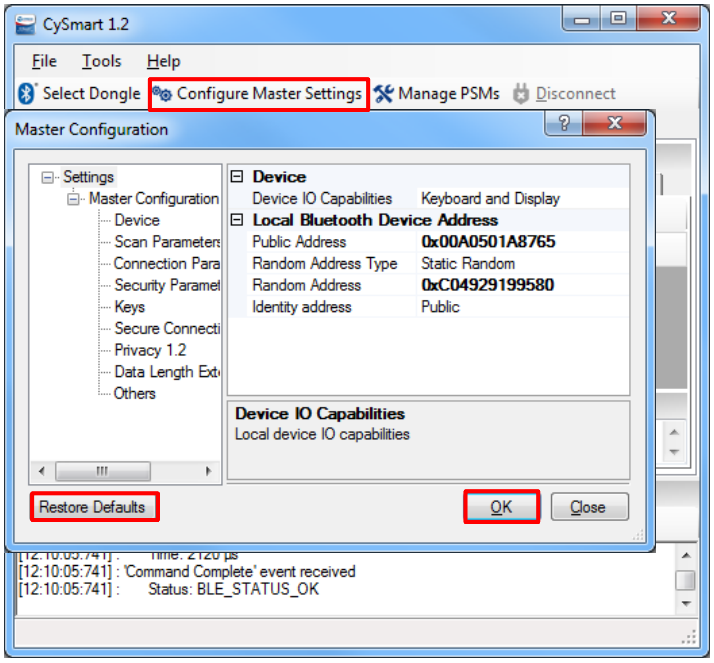
Figure 1. CySmart BLE Dongle Selection



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

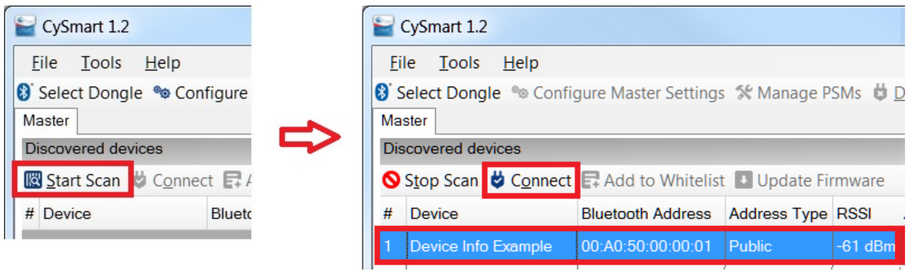
Device IO Capabilities
Local device IO capabilities

```
[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 the device is in the 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 **Device Info Example**) 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.

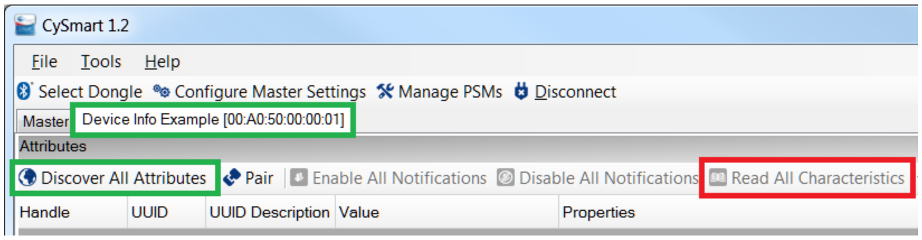
Figure 3. CySmart Device Discovery and Connection



#	Device	Bluetooth Address	Address Type	RSSI
1	Device Info Example	00:A0:50:00:00:01	Public	-61 dBm

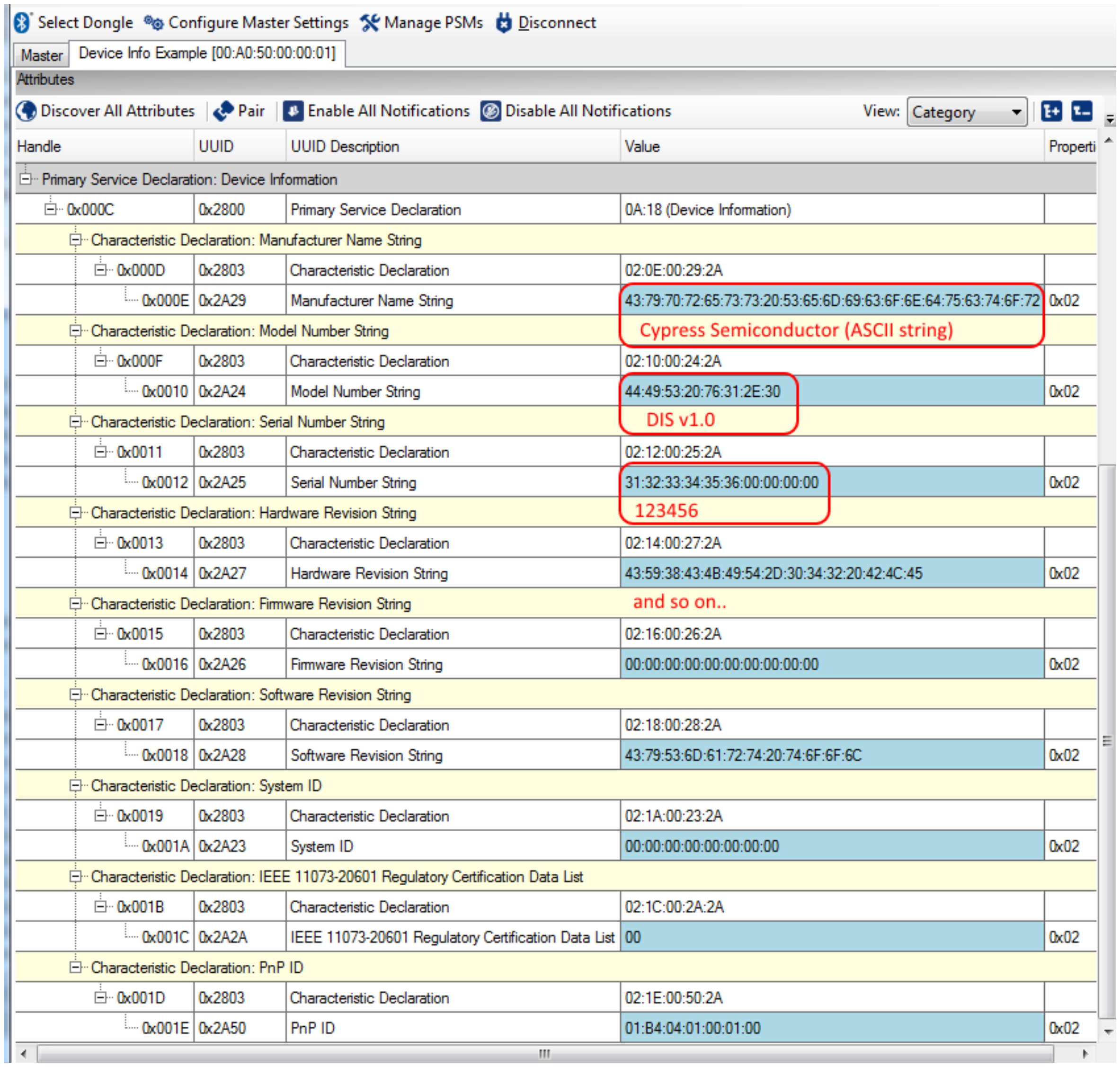
g. Once connected, switch to the **Device Info Example <device>** tab and click **Discover All Attributes** on your design from the CySmart Host Emulation Tool, as shown in Figure 4. Then, click **Read All Characteristics**.

Figure 4. CySmart Attribute Discovery and Characteristics Read



h. Observe the Device Information Service characteristics values read with example data, as shown in Figure 5.

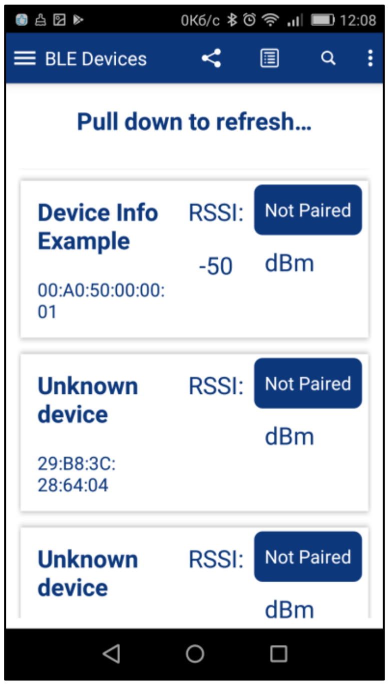
Figure 5. CySmart Device Information Service Characteristics Values



Handle	UUID	UUID Description	Value	Properti
Primary Service Declaration: Device Information				
0x000C	0x2800	Primary Service Declaration	0A:18 (Device Information)	
Characteristic Declaration: Manufacturer Name String				
0x000D	0x2803	Characteristic Declaration	02:0E:00:29:2A	
0x000E	0x2A29	Manufacturer Name String	43:79:70:72:65:73:73:20:53:65:6D:69:63:6F:6E:64:75:63:74:6F:72	0x02
Characteristic Declaration: Model Number String			Cypress Semiconductor (ASCII string)	
0x000F	0x2803	Characteristic Declaration	02:10:00:24:2A	
0x0010	0x2A24	Model Number String	44:49:53:20:76:31:2E:30	0x02
Characteristic Declaration: Serial Number String			DIS v1.0	
0x0011	0x2803	Characteristic Declaration	02:12:00:25:2A	
0x0012	0x2A25	Serial Number String	31:32:33:34:35:36:00:00:00:00	0x02
Characteristic Declaration: Hardware Revision String			123456	
0x0013	0x2803	Characteristic Declaration	02:14:00:27:2A	
0x0014	0x2A27	Hardware Revision String	43:59:38:43:4B:49:54:2D:30:34:32:20:42:4C:45	0x02
Characteristic Declaration: Firmware Revision String			and so on..	
0x0015	0x2803	Characteristic Declaration	02:16:00:26:2A	
0x0016	0x2A26	Firmware Revision String	00:00:00:00:00:00:00:00:00:00	0x02
Characteristic Declaration: Software Revision String				
0x0017	0x2803	Characteristic Declaration	02:18:00:28:2A	
0x0018	0x2A28	Software Revision String	43:79:53:6D:61:72:74:20:74:6F:6F:6C	0x02
Characteristic Declaration: System ID				
0x0019	0x2803	Characteristic Declaration	02:1A:00:23:2A	
0x001A	0x2A23	System ID	00:00:00:00:00:00:00:00	0x02
Characteristic Declaration: IEEE 11073-20601 Regulatory Certification Data List				
0x001B	0x2803	Characteristic Declaration	02:1C:00:2A:2A	
0x001C	0x2A2A	IEEE 11073-20601 Regulatory Certification Data List	00	0x02
Characteristic Declaration: PnP ID				
0x001D	0x2803	Characteristic Declaration	02:1E:00:50:2A	
0x001E	0x2A50	PnP ID	01:B4:04:01:00:01:00	0x02

5. Do the following to test the example using the CySmart mobile app as Running Speed and Cadence Service Client:
   a. Launch the CySmart mobile app and swipe down the screen to refresh the list of BLE devices available nearby. Make sure that the development kit is advertising (green LED is blinking): you may need to press the **SW1** button to wake up the device from the Hibernate mode. Once the **Device Info Example** device appears on the BLE devices list, connect to it and choose **Device Information Service** in the service selector.
   b. Explore information about the device.

Figure 6. CySmart Android App Recognized BLE Kit as Device Info Example



Pull down to refresh...

Device Info Example 00:A0:50:00:00:01	RSSI: -50	Not Paired dBm
Unknown device 29:B8:3C:28:64:04	RSSI:	Not Paired dBm
Unknown device	RSSI:	Not Paired dBm

Figure 7. CySmart Android App Shows Device Information Service in Service Selector

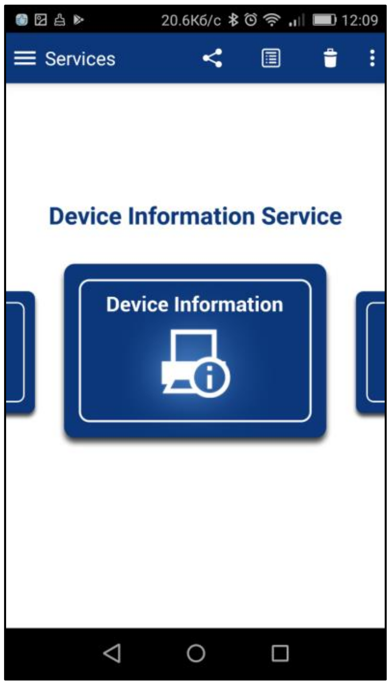
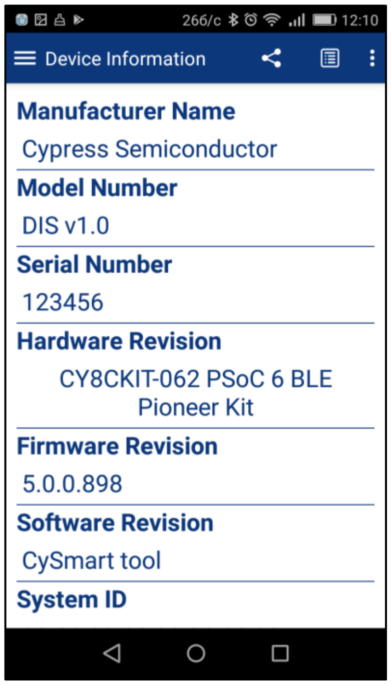


Figure 8. CySmart Android App Reads Device Information Service Characteristics

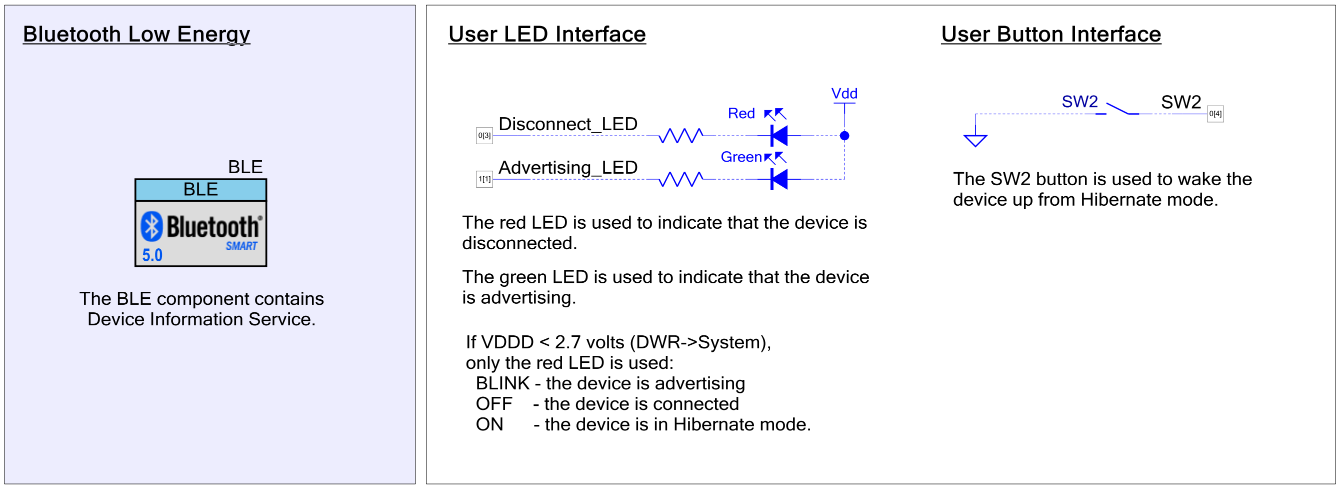


Manufacturer Name	Cypress Semiconductor
Model Number	DIS v1.0
Serial Number	123456
Hardware Revision	CY8CKIT-062 PSoC 6 BLE Pioneer Kit
Firmware Revision	5.0.0.898
Software Revision	CySmart tool
System ID	

Design and Implementation

Figure 9 shows the top design schematic.

Figure 9. BLE Device Information Service Code Example Schematic



The BLE component contains Device Information Service.

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 wake the device up from Hibernate mode.

One callback function (`AppCallBack()`) is required to receive generic events from the BLE Stack.

The `Cy_BLE_GAPP_StartAdvertisement()` API is called after the `CY_BLE_EVT_STACK_ON` event to start advertising.

Pin assignments

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

Table 1. Pin Assignment

Pin Name	Development Kit CY8CKIT-062	Comment
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

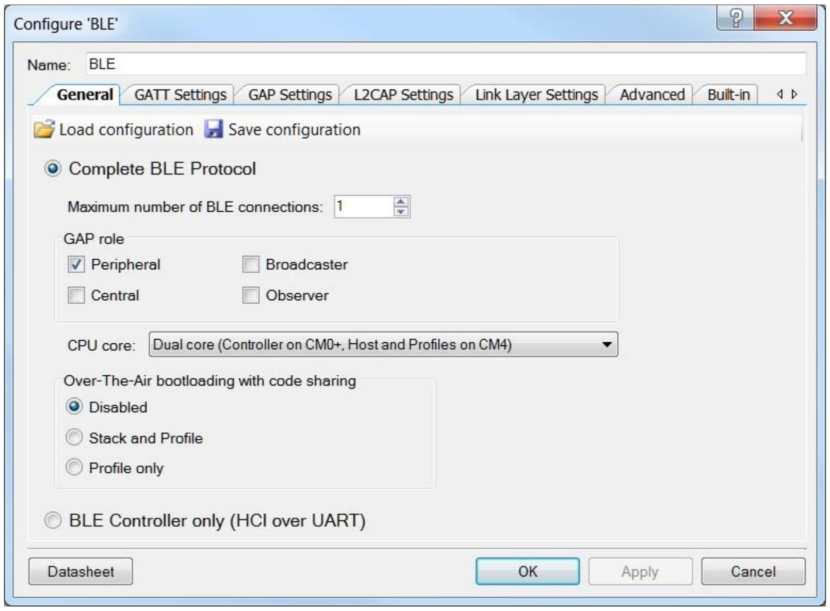
Component	Instance Name	Purpose	Non-default Settings
Bluetooth Low Energy (BLE)	BLE	The BLE component contains Device Information Service.	See the Parameter Settings section
Digital Input Pin	SW2	The **SW2** button is used to wake the device up from Hibernate mode.	**[General tab]** Uncheck HW connection 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

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

Parameter Settings

The BLE component is configured as the custom profile in peripheral role. Device Information Service is added with all characteristics.

Figure 10. General Settings



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

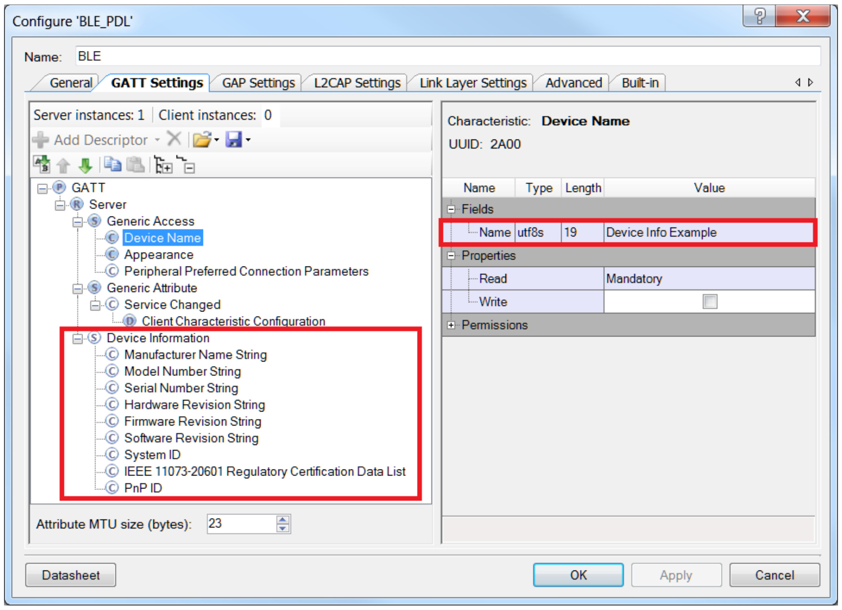
[✓ Disabled]

Stack and Profile

Profile only

BLE Controller only (HCI over UART)

Figure 11. GATT Settings


Configure 'BLE_PDL'

Name: BLE

General | **GATT Settings** | GAP Settings | L2CAP Settings | Link Layer Settings | Advanced | Built-in

Server instances: 1 | Client instances: 0

Add Descriptor

- GATT
  - Server
    - Generic Access
      - Device Name
      - Appearance
      - Peripheral Preferred Connection Parameters
    - Generic Attribute
      - Service Changed
        - Client Characteristic Configuration
    - Device Information
      - Manufacturer Name String
      - Model Number String
      - Serial Number String
      - Hardware Revision String
      - Firmware Revision String
      - Software Revision String
      - System ID
      - IEEE 11073-20601 Regulatory Certification Data List
      - PnP ID

Attribute MTU size (bytes): 23

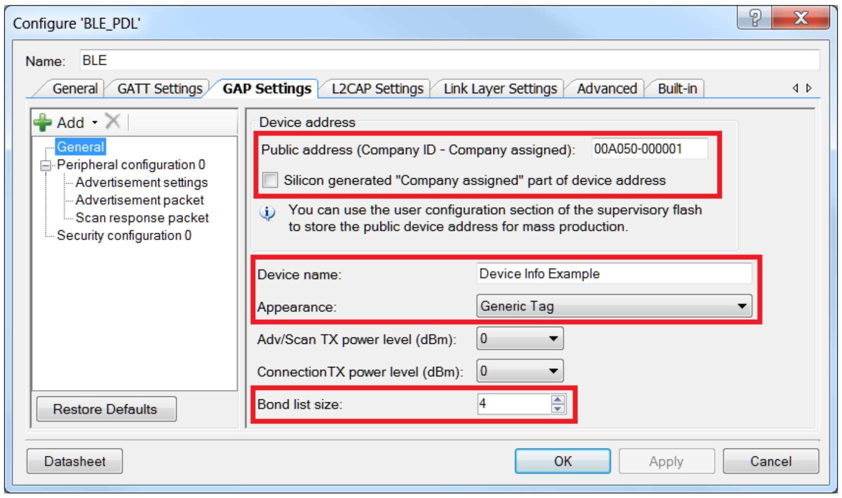
Characteristic: **Device Name**

UUID: 2A00

Name	Type	Length	Value
Fields			
Name	utf8s	19	Device Info Example
Properties			
Read			Mandatory
Write			☐
Permissions			

Datasheet | OK | Apply | Cancel

Figure 12. GAP Settings


Configure 'BLE_PDL'

Name: BLE

General | GATT Settings | **GAP Settings** | L2CAP Settings | Link Layer Settings | Advanced | Built-in

Add

- General
- Peripheral configuration 0
  - Advertisement settings
  - Advertisement packet
  - Scan response packet
- Security configuration 0

Restore Defaults

Device address

Public address (Company ID - Company assigned): 00A050-000001

☐ 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: Device Info Example

Appearance: Generic Tag

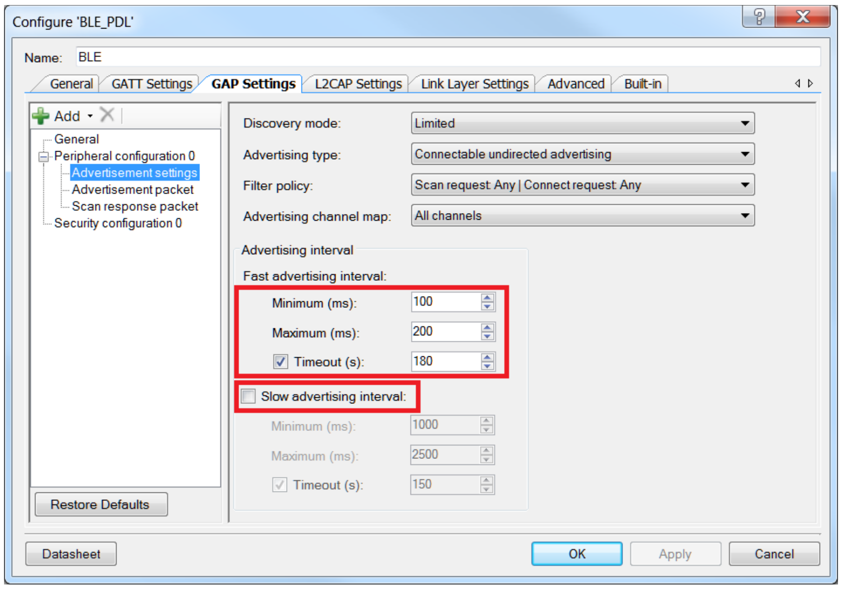
Adv/Scan TX power level (dBm): 0

ConnectionTX power level (dBm): 0

Bond list size: 4

Datasheet | OK | Apply | Cancel

Figure 13. GAP Settings: Advertisement Settings



Configure 'BLE_PDL'

Name: BLE

General | GATT Settings | **GAP Settings** | L2CAP Settings | Link Layer Settings | Advanced | Built-in

- General
- Peripheral configuration 0
  - Advertisement settings
  - Advertisement packet
  - Scan response packet
- Security configuration 0

Discovery mode:	Limited
Advertising type:	Connectable undirected advertising
Filter policy:	Scan request: Any | Connect request: Any
Advertising channel map:	All channels

Advertising interval

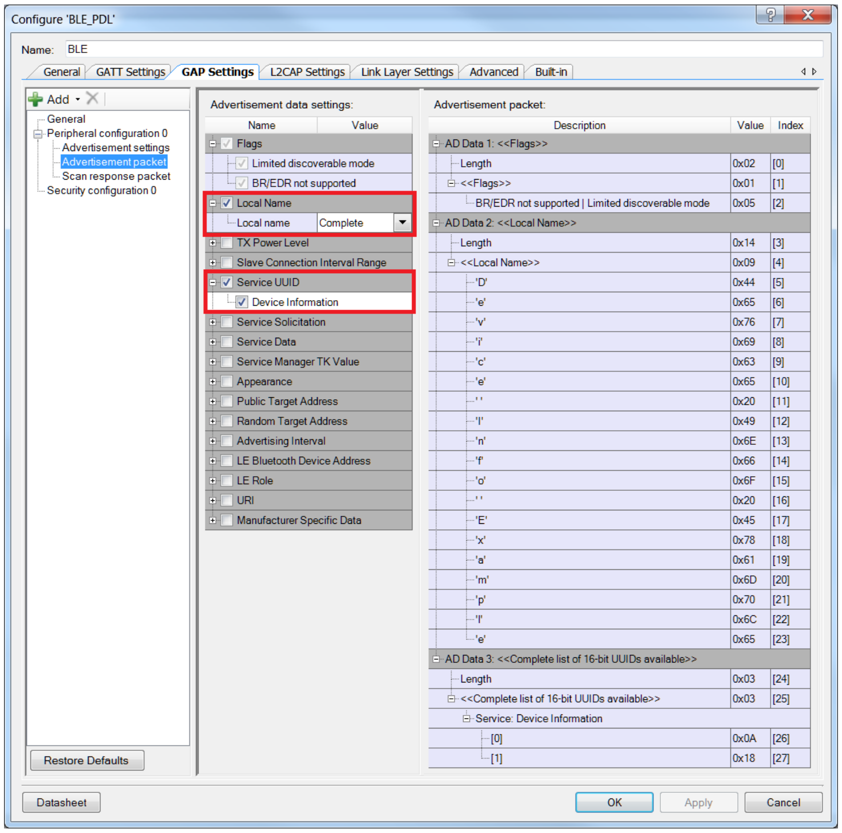
Fast advertising interval:

Minimum (ms):	100
Maximum (ms):	200
✓ Timeout (s):	180

☐ Slow advertising interval:

Minimum (ms):	1000
Maximum (ms):	2500
✓ Timeout (s):	150

Figure 14. GAP Settings: Advertisement Packet



Configure 'BLE_PDL'

Name: BLE

General | GATT Settings | **GAP Settings** | L2CAP Settings | Link Layer Settings | Advanced | Built-in

- General
- Peripheral configuration 0
  - Advertisement settings
  - Advertisement packet
  - Scan response packet
- Security configuration 0

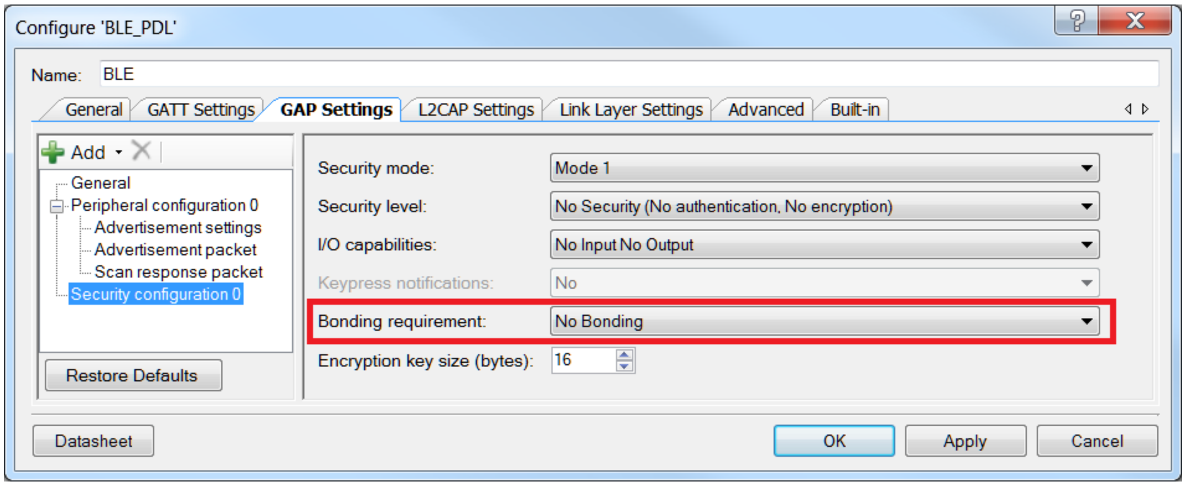
Advertisement data settings:

Name	Value
✓ Flags	
✓ Limited discoverable mode	
✓ BR/EDR not supported	
✓ Local Name	
Local name	Complete
☐ TX Power Level	
☐ Slave Connection Interval Range	
✓ Service UUID	
✓ Device Information	
☐ Service Solicitation	
☐ Service Data	
☐ 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	0x14	[3]
<<Local Name>>	0x09	[4]
'D'	0x44	[5]
'e'	0x65	[6]
'v'	0x76	[7]
'i'	0x69	[8]
'c'	0x63	[9]
'e'	0x65	[10]
' '	0x20	[11]
'I'	0x49	[12]
'n'	0x6E	[13]
'f'	0x66	[14]
'o'	0x6F	[15]
' '	0x20	[16]
'E'	0x45	[17]
'x'	0x78	[18]
'a'	0x61	[19]
'm'	0x6D	[20]
'p'	0x70	[21]
'l'	0x6C	[22]
'e'	0x65	[23]
AD Data 3: <<Complete list of 16-bit UUIDs available>>		
Length	0x03	[24]
<<Complete list of 16-bit UUIDs available>>	0x03	[25]
Service: Device Information		
[0]	0x0A	[26]
[1]	0x18	[27]

Figure 15. Security Settings



Name: BLE

Security mode: Mode 1

Security level: No Security (No authentication, No encryption)

I/O capabilities: No Input No Output

Keypress notifications: No

Bonding requirement: No Bonding

Encryption key size (bytes): 16

Switching the CPU Cores Usage

This section describes how to switch between different CPU cores usage (Single core and Dual core) in the BLE Peripheral Driver Library (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+ will be used.
- **Single core (Complete Component on CM4)** – only CM4 will be used.
- **Dual core (Controller on CM0+, Host and Profiles on CM4)** – CM0+ and CM4 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.

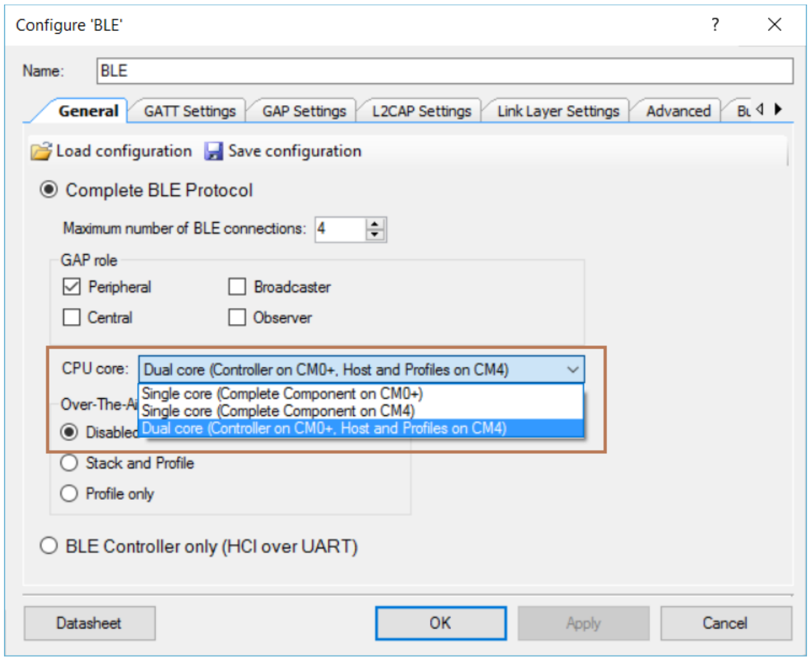
Important to remember:

- All application host-files must be run on the host core.
- The BLE Subsystem (BLESS) interrupt must be assigned to the core where the controller runs.

Do the following to switch the CPU cores usage:

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

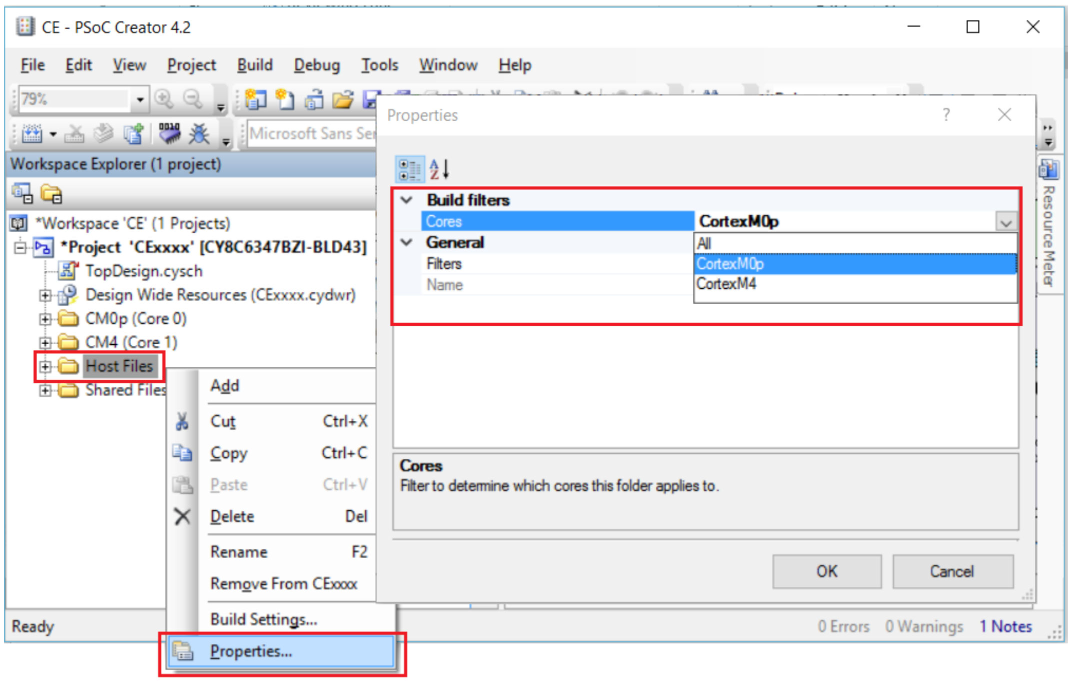
Figure 16. Select CPU Core


Configure 'BLE'
Name: BLE
General
GATT Settings
GAP Settings
L2CAP Settings
Link Layer Settings
Advanced
Load configuration
Save configuration
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)
Datasheet
OK
Apply
Cancel

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 17.

   - 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 17. Change Core Properties


CE - PSoC Creator 4.2
File Edit View Project Build Debug Tools Window Help
Workspace Explorer (1 project)
*Workspace 'CE' (1 Projects)
*Project 'CExxxx' [CY8C6347BZI-BLD43]
TopDesign.cysch
Design Wide Resources (CExxxx.cydwr)
CM0p (Core 0)
CM4 (Core 1)
Host Files
Shared Files
Add
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Rename F2
Remove From CExxxx
Build Settings...
Properties...
Properties
OK
Cancel
Ready
0 Errors 0 Warnings 1 Notes

Build filters	
Cores	CortexM0p
General	
Filters	
Name	

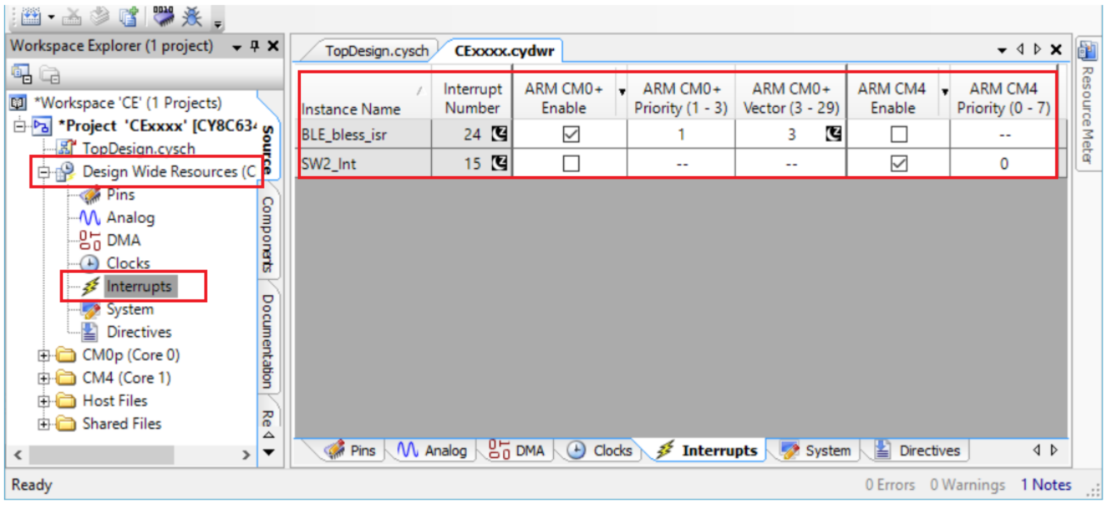
All
CortexM0p
CortexM4

Cores
Filter to determine which cores this folder applies to.

3. Assign the BLE_bless_isr and other peripheral (button – SW2, timer(s), and so on) interrupts to the 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 18. 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, kit, or both, change the target device using the Device Selector and update the pin assignments in the Design Wide Resources Pins settings as needed.

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: CE215120 - BLE Device Information Service with PSoC 6 MCU with BLE Connectivity

Document Number: 002-15120

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6086706	NPAL	06/12/2018	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 | 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