CE217642 - BLE Running Speed and Cadence Profile with PSoC 6 MCU with BLE Connectivity

Objective

This example project demonstrates the Running Speed and Cadence Sensor application workflow.

Overview

The design demonstrates the core functionality of the BLE Component configured as the Running Speed and Cadence Sensor (RSCS) device in the GATT Server role. The device simulates running/walking data measurements and sends them over the BLE Running Speed and Cadence Service.

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

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.

LED behavior for VDDD Voltage > 2.7 volts is described in **Operation** section.

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 project is intended to work with any BLE-compatible device (e.g. phone, tablet). The appropriate software with the Running Speed and Cadence Profile support should be installed on the Client OS. After the Client device is connected to the Running Speed and Cadence Sensor, the Client device enables notifications for the RSC Measurement Characteristic and the Sensor is sending notifications to the peer Client periodically. The period of the notifications is 3 seconds. The notifications sent from the Sensor contain “Instantaneous Speed”, “Instantaneous Cadence”, “Instantaneous Stride Length”, “Total Distance”, and “Walking or Running Status bit” fields.

The example project also periodically increases the value of “Instantaneous Cadence”, “Instantaneous Stride Length”, and, therefore, “Instantaneous Speed”. The increase period of values is 10 seconds. After the “Instantaneous Cadence” and “Instantaneous Stride Length” reaches their maximum allowed values, they will be reset back to the minimum values. By default, the project simulates the “Walking” profile, but it also can simulate the “Running” profile.

To switch to the “Running” profile, press and hold the **SW2** button on the CY8CKIT-062 PSoC® 6 BLE Pioneer Kit. When the “Running” profile becomes active, the blue LED is turned ON indicating this. While connected to the Client and between the connection intervals, the device is put into low-power mode.

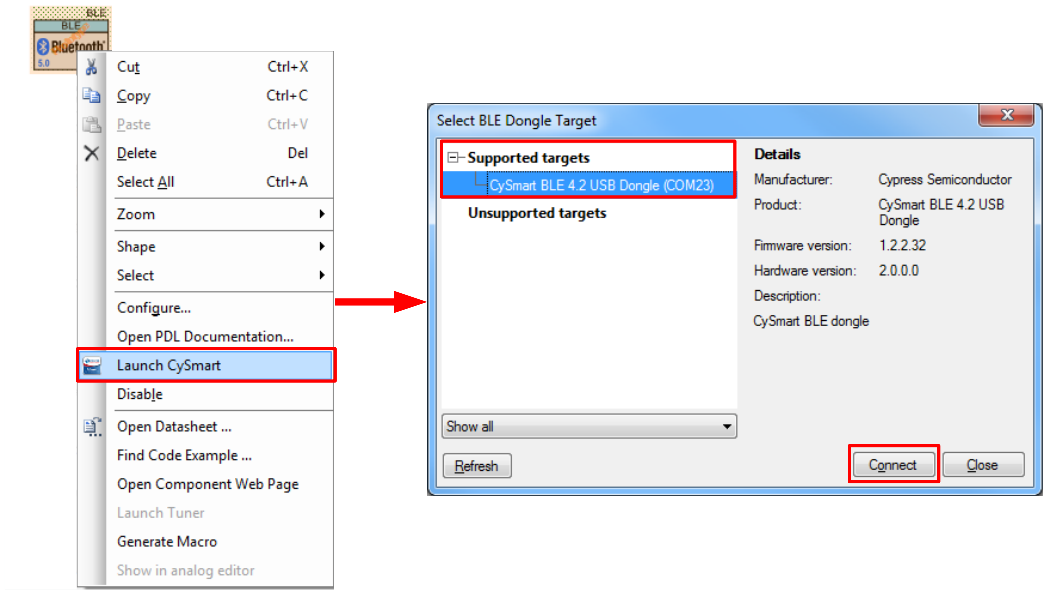
Refer to the Running Speed and Cadence Profile and Running Speed and Cadence Service specifications for more details.

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 (VDDD configuration > 2.7) 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 Running Speed and Cadence Service Client:
    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



Details

Manufacturer:	Cypress Semiconductor
Product:	CySmart BLE 4.2 USB Dongle
Firmware version:	1.2.2.32
Hardware version:	2.0.0.0
Description:	

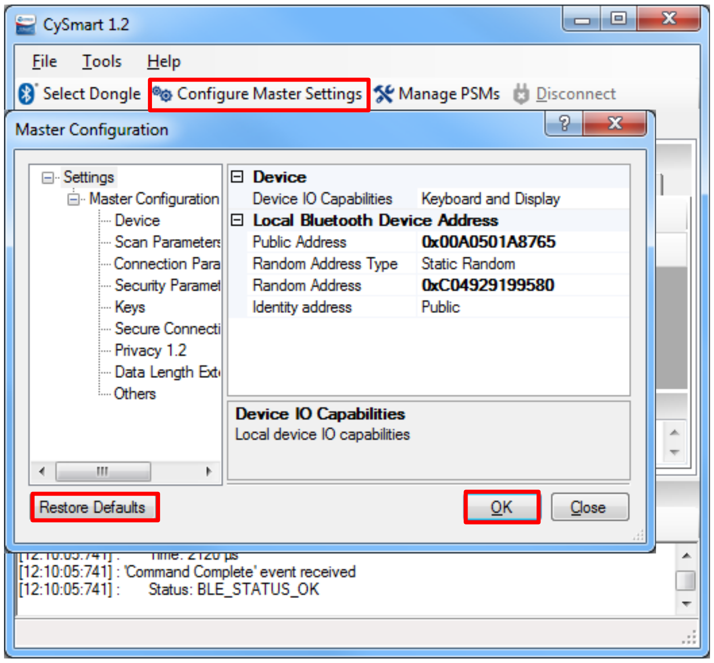
CySmart BLE dongle

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

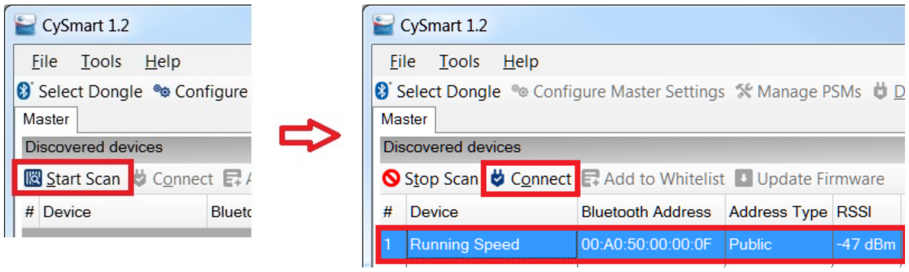
Device IO Capabilities
Local device IO capabilities

```
[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.

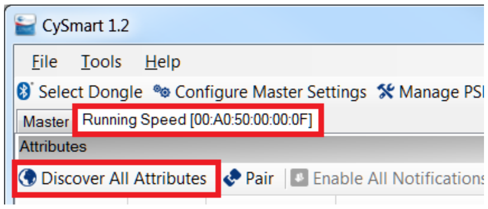
f. On the CySmart Host Emulation Tool, click **Start Scan**. Your device name (configured as **Running Speed and Cadence Sensor**) should appear in the Discovered devices list, as Figure 3 shows. Select the device and click **Connect** to establish a BLE connection between the CySmart Host Emulation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

Figure 3. CySmart Device Discovery and Connection


CySmart 1.2
File Tools Help
Select Dongle Configure
Master
Discovered devices
Start Scan Connect
# Device Blueto
CySmart 1.2
File Tools Help
Select Dongle Configure Master Settings Manage PSMs D
Master
Discovered devices
Stop Scan Connect Add to Whitelist Update Firmware
# Device Bluetooth Address Address Type RSSI
1 Running Speed 00:A0:50:00:00:0F Public -47 dBm

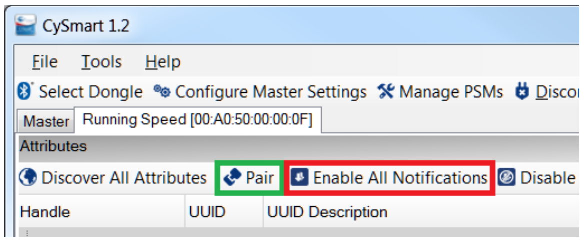
g. Once connected, switch to the '**Running Speed**' device tab and '**Discover all Attributes**' on your design from the CySmart Host Emulation Tool, as shown in Figure 4.

Figure 4. CySmart Attribute Discovery


CySmart 1.2
File Tools Help
Select Dongle Configure Master Settings Manage PSI
Master Running Speed [00:A0:50:00:00:0F]
Attributes
Discover All Attributes Pair Enable All Notifications

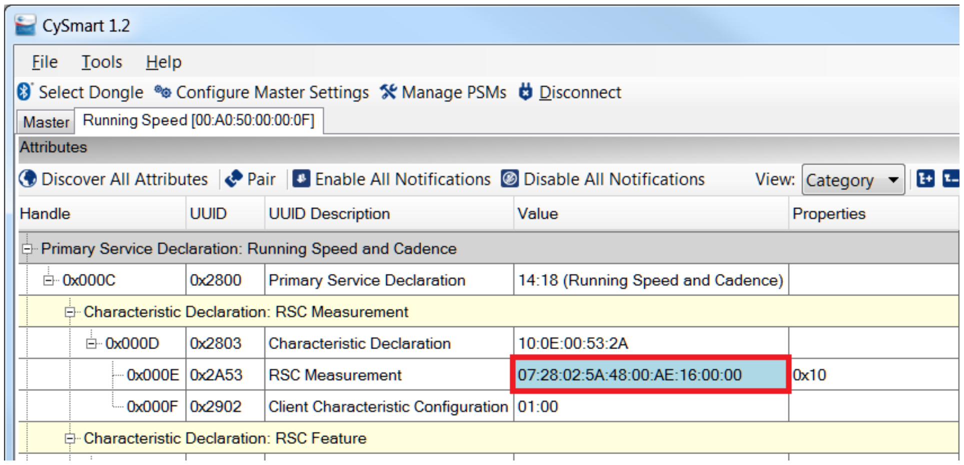
h. Click **Pair** after discovery finishes, then **Enable All Notifications** in the CySmart app as shown in Figure 5.

Figure 5. CySmart Pair and Enable All Notification


CySmart 1.2
File Tools Help
Select Dongle Configure Master Settings Manage PSMs Disco
Master Running Speed [00:A0:50:00:00:0F]
Attributes
Discover All Attributes Pair Enable All Notifications Disable
Handle UUID UUID Description

i. Observe the Running Speed Reporter start notification of the RSC Measurement Characteristic:

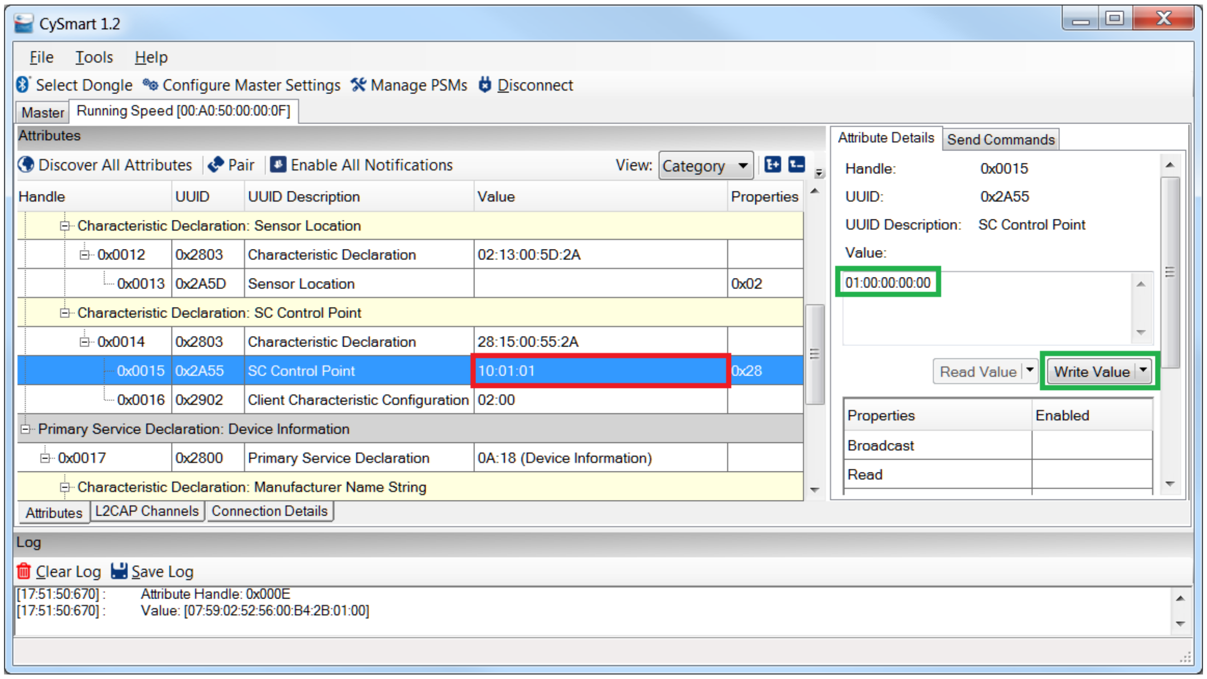
Figure 6. RSC Measurement Characteristic Notification


CySmart 1.2
File Tools Help
Select Dongle Configure Master Settings Manage PSMs Disconnect
Master Running Speed [00:A0:50:00:00:0F]
Attributes
Discover All Attributes Pair Enable All Notifications Disable All Notifications View: Category

Handle	UUID	UUID Description	Value	Properties
Primary Service Declaration: Running Speed and Cadence				
0x000C	0x2800	Primary Service Declaration	14:18 (Running Speed and Cadence)	
Characteristic Declaration: RSC Measurement				
0x000D	0x2803	Characteristic Declaration	10:0E:00:53:2A	
0x000E	0x2A53	RSC Measurement	07:28:02:5A:48:00:AE:16:00:00	0x10
0x000F	0x2902	Client Characteristic Configuration	01:00	
Characteristic Declaration: RSC Feature				

j. Send the Set Cumulative Value Op Code to the SC Control Point. Select the SC Control Point and write the 01:XX:XX:XX:XX, where 01 is a Set Cumulative Procedure Op Code XX:XX:XX:XX is a 4-byte cumulative value, then press **Write Value**. Observe a response from the SC Control Point. The general format of a response: XX:YY:ZZ, where XX – the response Op Code, YY – the requested Op Code, ZZ – the status byte. The status byte can be set to one of the following values: 0x01 – Success, 0x02 – Op Code is not supported, 0x03 – Invalid parameter, 04 – Operation failed.

Figure 7. Set Cumulative Value Procedure Response



Handle	UUID	UUID Description	Value	Properties
Characteristic Declaration: Sensor Location				
0x0012	0x2803	Characteristic Declaration	02:13:00:5D:2A	
0x0013	0x2A5D	Sensor Location		0x02
Characteristic Declaration: SC Control Point				
0x0014	0x2803	Characteristic Declaration	28:15:00:55:2A	
0x0015	0x2A55	SC Control Point	10:01:01	0x28
0x0016	0x2902	Client Characteristic Configuration	02:00	
Primary Service Declaration: Device Information				
0x0017	0x2800	Primary Service Declaration	0A:18 (Device Information)	
Characteristic Declaration: Manufacturer Name String				

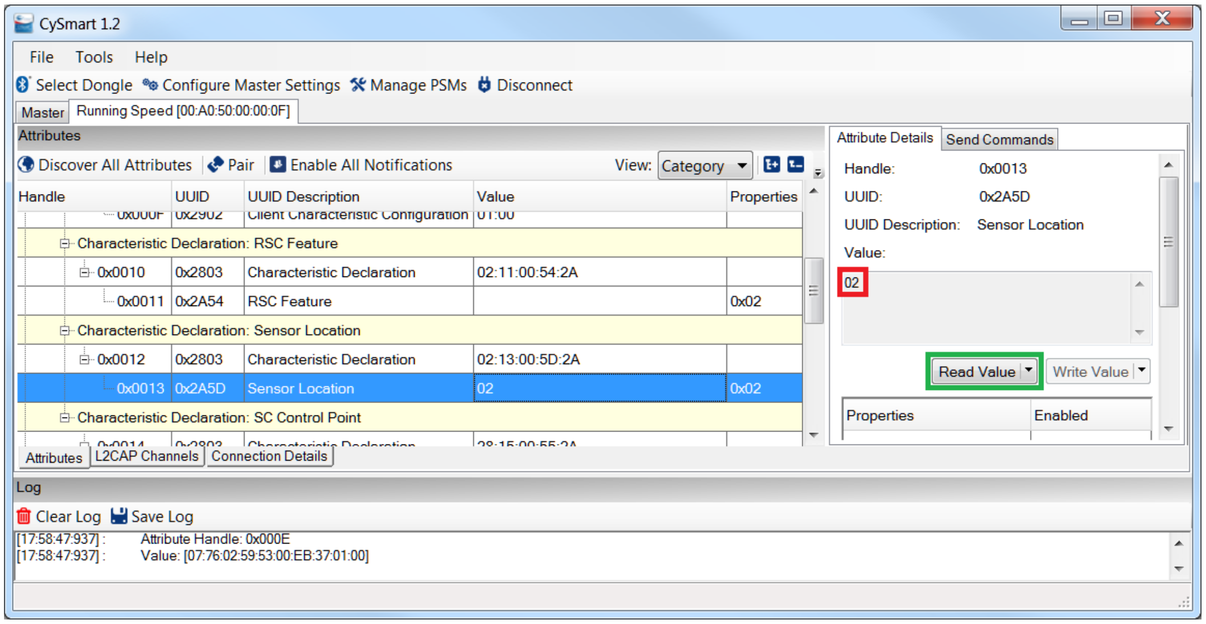
Handle: 0x0015
UUID: 0x2A55
UUID Description: SC Control Point
Value:
01:00:00:00:00

Properties	Enabled
Broadcast	
Read	

```
[17:51:50:670] :    Attribute Handle: 0x000E
[17:51:50:670] :    Value: [07:59:02:52:56:00:B4:2B:01:00]
```

k. Select the Sensor Location Characteristic and press **Read Value**. Observe the sensor location value:

Figure 8. Read Sensor Location Characteristic



Handle	UUID	UUID Description	Value	Properties
0x000F	0x2902	Client Characteristic Configuration	01:00	
Characteristic Declaration: RSC Feature				
0x0010	0x2803	Characteristic Declaration	02:11:00:54:2A	
0x0011	0x2A54	RSC Feature		0x02
Characteristic Declaration: Sensor Location				
0x0012	0x2803	Characteristic Declaration	02:13:00:5D:2A	
0x0013	0x2A5D	Sensor Location	02	0x02
Characteristic Declaration: SC Control Point				
0x0014	0x2803	Characteristic Declaration	28:15:00:55:2A	

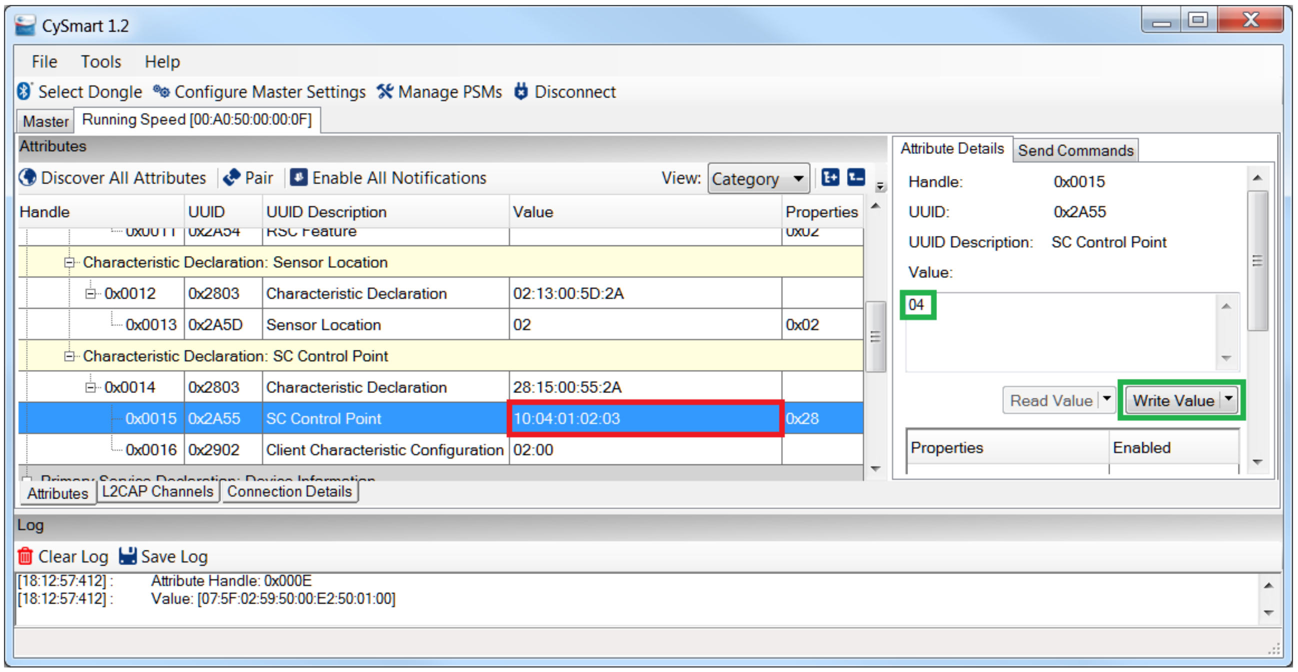
Handle: 0x0013
UUID: 0x2A5D
UUID Description: Sensor Location
Value:
02

Properties	Enabled

```
[17:58:47:937] :    Attribute Handle: 0x000E
[17:58:47:937] :    Value: [07:76:02:59:53:00:EB:37:01:00]
```

l. Send the Request Sensor Locations Op Code to the SC Control Point. Write 04 in the **Value** field, then press **Write Value**.

Figure 9. Request Sensor Locations Response



Handle	UUID	UUID Description	Value	Properties
0x0011	0x2A54	RSC Feature		0x02
Characteristic Declaration: Sensor Location				
0x0012	0x2803	Characteristic Declaration	02:13:00:5D:2A	
0x0013	0x2A5D	Sensor Location	02	0x02
Characteristic Declaration: SC Control Point				
0x0014	0x2803	Characteristic Declaration	28:15:00:55:2A	
0x0015	0x2A55	SC Control Point	10:04:01:02:03	0x28
0x0016	0x2902	Client Characteristic Configuration	02:00	

Handle:	0x0015
UUID:	0x2A55
UUID Description:	SC Control Point
Value:	04

Properties	Enabled

```
[18:12:57:412] :      Attribute Handle: 0x000E
[18:12:57:412] :      Value: [07:5F:02:59:50:00:E2:50:01:00]
```

m. Observe a response from the SC Control Point. If the Request Sensor Locations procedure is successful, the first three bytes of the response will have the same meaning as for the Set Cumulative Value procedure and all the remaining bytes are identifying the supported sensor locations. If the procedure fails, there will be only three bytes in a response.

n. Send the Update Sensor Location Op Code to the SC Control Point. Write 03:03 in the **Value** field then press **Write Value** and observe a response. Read the Sensor Location Characteristic value again and observe that the new sensor location was set.

6. Do the following to test example, using the CySmart mobile app as Running Speed and Cadence Service Client:

   a. Launch 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 in order to wake up the device from Hibernate mode. Once the “Running Speed” device appears on the BLE devices list, connect to it and choose “Running Speed & Cadence Service” in the service selector.

Figure 10. CySmart iOS App Recognized BLE Kit as Running Speed Reporter

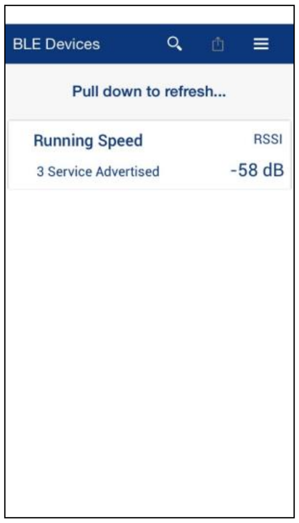
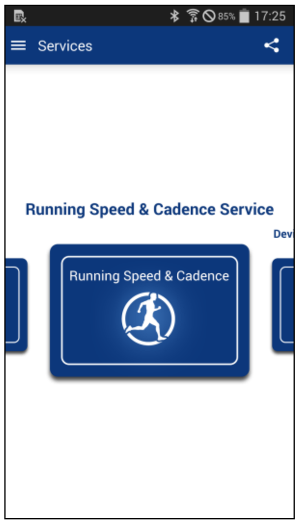


Figure 11. CySmart Android App Shows "Running Speed & Cadence Service" in Service Selector



b. Once connected, the CySmart mobile application provides an interface for measurement of a distance being run, calories burnt and average speed based on the running speed and cadence values obtained from the development kit. To attain this functionality, enter the runner's weight into the appropriate textbox and press the **Start** button:

Figure 12. Running Speed and Cadence Service Interface on CySmart App Provides Possibility to Enter Runner's Weight

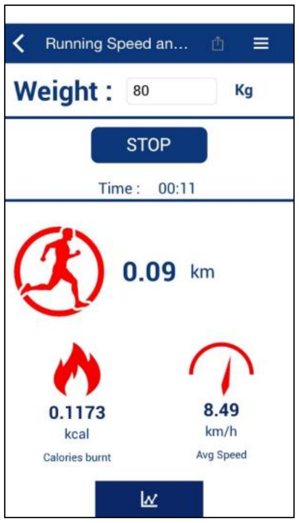
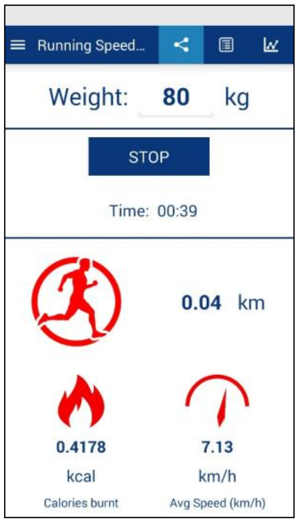


Figure 13. Cysmart Android App Simulates Measurement of Total Distance Being Run, amount of Calories Burnt and Average Speed



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.

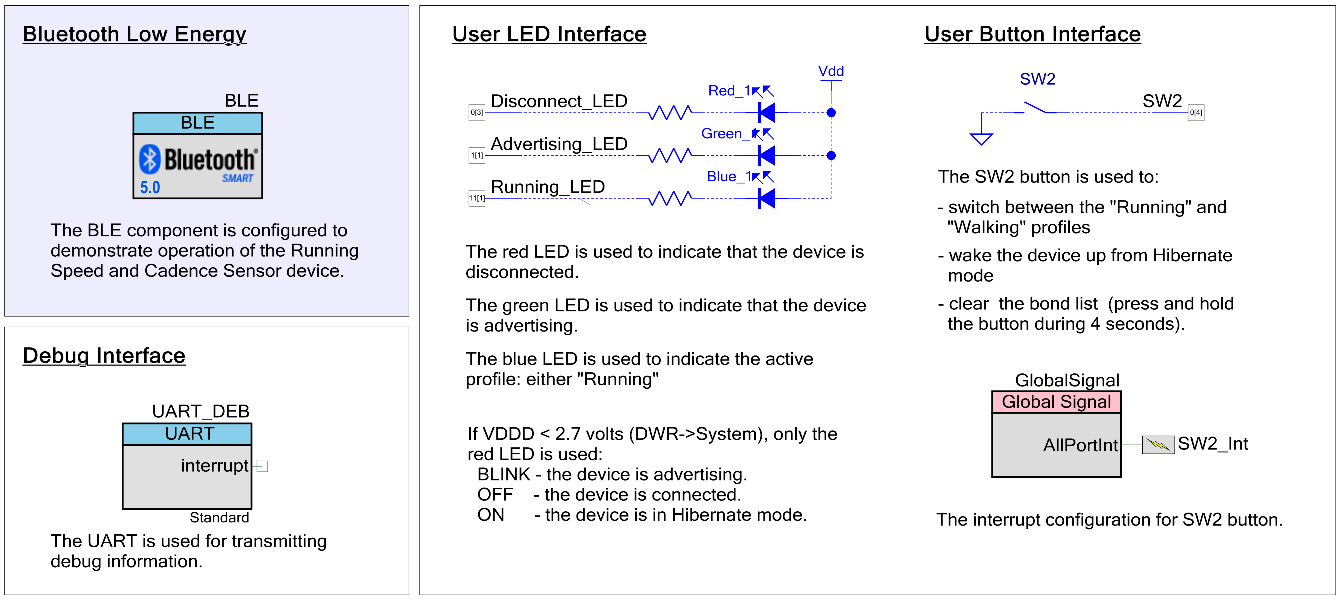
BLE Running Speed and Cadence Profile Example Project
CY_BLE_EVT_STACK_ON, StartAdvertisement
CY_BLE_EVT_SET_DEVICE_ADDR_COMPLETE
CY_BLE_EVT_LE_SET_EVENT_MASK_COMPLETE

```
CY_BLE_EVT_GET_DEVICE_ADDR_COMPLETE: 00a05000000f
CY_BLE_EVT_SET_TX_PWR_COMPLETE
CY_BLE_EVT_SET_TX_PWR_COMPLETE
CY_BLE_EVT_GAPP_ADVERTISEMENT_START_STOP, state: 2
CY_BLE_EVT_GAP_KEYS_GEN_COMPLETE
CY_BLE_EVT_GATT_CONNECT_IND: 0, 10
CY_BLE_EVT_GAP_DEVICE_CONNECTED: connIntv = 7 ms
CY_BLE_EVT_GATTS_XCNHG_MTU_REQ
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ: handle: 3
CY_BLE_EVT_GAP_AUTH_REQ: bdHandle=10, security=3, bonding=1, ekeySize=10, err=0
CY_BLE_EVT_GAP_SMP_NEGOTIATED_AUTH_INFO: bdHandle=10, security=1, bonding=1, ekeySize=10, err=0
CY_BLE_EVT_STACK_BUSY_STATUS: 1
CY_BLE_EVT_GAP_ENCRYPT_CHANGE: 0
CY_BLE_EVT_STACK_BUSY_STATUS: 0
CY_BLE_EVT_GAP_KEYINFO_EXCHNGE_CMPLT
CY_BLE_EVT_GAP_AUTH_COMPLETE: security:1, bonding:1, ekeySize:10, authErr 0
CY_BLE_EVT_PENDING_FLASH_WRITE
Store bonding data, status: 140001, pending: 1
Store bonding data, status: 140001, pending: 1
Store bonding data, status: 140001, pending: 1
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_INDICATION_ENABLED
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ: handle: b
Indications for SC Control point Characteristic are enabled
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ: handle: 16
Notifications for RSC Measurement Characteristic are enabled
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ: handle: f
Notification is sent!
Cadence: 81, Speed: 5.92 km/h, Stride length: 61 cm, Total distance: 6 m, Status: Walking
Notification is sent!
Cadence: 81, Speed: 5.92 km/h, Stride length: 61 cm, Total distance: 8 m, Status: Walking
Notification is sent!
Cadence: 81, Speed: 5.92 km/h, Stride length: 61 cm, Total distance: 10 m, Status: Walking
Notification is sent!
Cadence: 82, Speed: 6.08 km/h, Stride length: 62 cm, Total distance: 12 m, Status: Walking
Notification is sent!
Cadence: 82, Speed: 6.08 km/h, Stride length: 62 cm, Total distance: 13 m, Status: Walking
Notification is sent!
Cadence: 82, Speed: 6.08 km/h, Stride length: 62 cm, Total distance: 15 m, Status: Walking
Notification is sent!
Cadence: 82, Speed: 6.08 km/h, Stride length: 62 cm, Total distance: 17 m, Status: Walking
Notification is sent!
Cadence: 83, Speed: 6.27 km/h, Stride length: 63 cm, Total distance: 19 m, Status: Walking
Notification is sent!
Cadence: 83, Speed: 6.27 km/h, Stride length: 63 cm, Total distance: 21 m, Status: Walking
Notification is sent!
Cadence: 83, Speed: 6.27 km/h, Stride length: 63 cm, Total distance: 23 m, Status: Walking
Notification is sent!
Cadence: 84, Speed: 6.44 km/h, Stride length: 64 cm, Total distance: 25 m, Status: Walking
Write to SC Control Point Characteristic occurred
Data length: 5
Received data: 1 0 0 0 0
Set cumulative value command was received.
Cy_BLE_RSCSS_SendIndication() succeeded
Indication Confirmation for SC Control point was received
Notification is sent!
Cadence: 86, Speed: 6.80 km/h, Stride length: 66 cm, Total distance: 1 m, Status: Walking
Notification is sent!
Cadence: 86, Speed: 6.80 km/h, Stride length: 66 cm, Total distance: 3 m, Status: Walking
Notification is sent!
Cadence: 86, Speed: 6.80 km/h, Stride length: 66 cm, Total distance: 5 m, Status: Walking
Notification is sent!
Cadence: 87, Speed: 6.98 km/h, Stride length: 67 cm, Total distance: 7 m, Status: Walking
```

Design and Implementation

The project demonstrates the core functionality of the BLE Component configured as the Running Speed and Cadence Sensor (RSCS) device in the GATT Server role. The device simulates running/walking data measurements and sends them over the BLE Running Speed and Cadence Service. Figure 14 shows the top design schematic.

Figure 14. BLE Running Speed and Cadence Profile Code-Example Schematic



The BLE component is configured to demonstrate operation of the Running Speed and Cadence Sensor 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.

The blue LED is used to indicate the active profile: either "Running"

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 the "Running" and "Walking" profiles
- wake the device up from Hibernate mode
- clear the bond list (press and hold the button during 4 seconds).

The interrupt configuration for SW2 button.

For operation the example project uses two callback functions: AppCallBack() and RscServiceAppEventHandler(). One callback function (AppCallBack()) is required for receiving generic events from the BLE Stack, and the second (RscServiceAppEventHandler()) is required for receiving events from the Running Speed and Cadence Service. The example project uses the UART Component for displaying debug information.

To start the example project operation, build it and program into the CY8CKIT-062 PSoC® 6 BLE Pioneer Kit. After a startup, the BLE, UART and ISR Components are initialized and the BLE Component begins its operation. The RGB LED starts blinking with green. This indicates that the device has started advertising and it is available for the connection with a Central device. To advertise, the example project uses the packet structure as configured in the BLE Component Customizer.

This example requires an external GAP Central device configured in the GATT Client role for operation. To connect to the Running Speed and Cadence Sensor device, send a connection request when the device is advertising. After a 30-second period, if no Central device has connected to the Running Speed and Cadence Sensor, the Sensor device will stop advertising and the red LED is turned ON indicating the disconnection state.

While connected to a 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 CY8CKIT-062	Comment
\UART_DEB:rx\	P5[0]	
\UART_DEB:tx\	P5[1]	
\UART_DEB:rts\	P5[2]	
\UART_DEB:cts\	P5[3]	
Advertising_LED	P1[1]	The green color of the RGB LED.

Pin Name	Development Kit CY8CKIT-062	Comment
Disconnect_LED	P0[3]	The red color of the RGB LED.
Running_LED	P11[1]	The blue 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

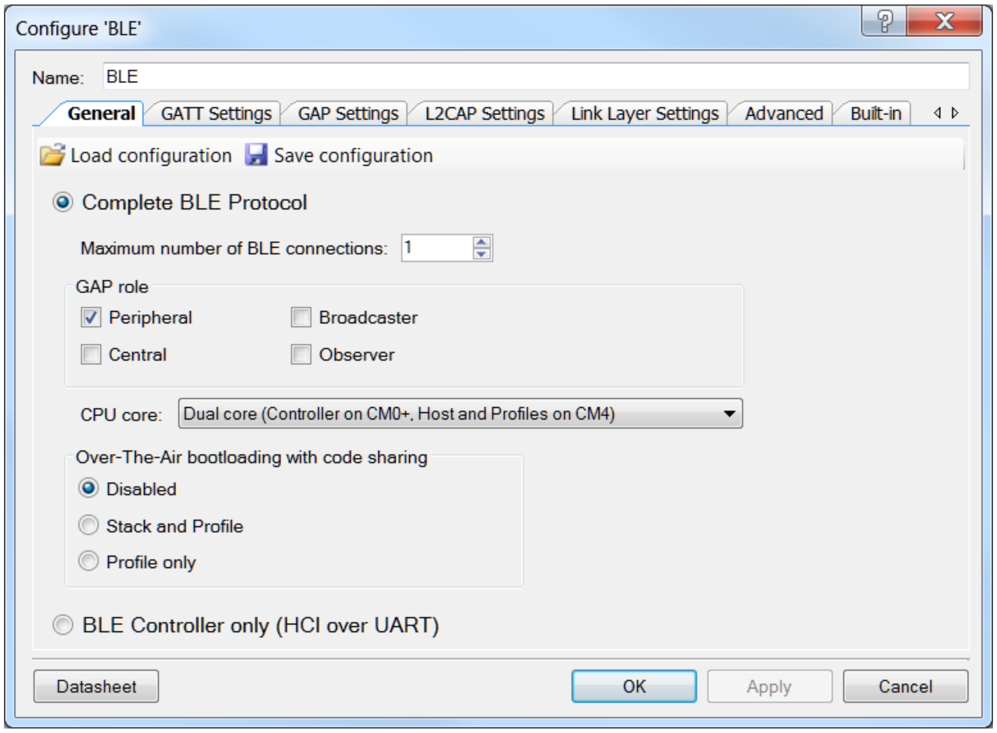
Component	Instance Name	Purpose	Non-default Settings
Bluetooth Low Energy (BLE)	BLE	The BLE component is configured to demonstrate operation of the Environmental Sensing Sensor device.	Refer to Parameter Settings section
Digital Input Pin	SW2	This pin is used to generate interrupts when the user button (**SW2**) is pressed.	**[General tab]** Uncheck HW connection Drive mode: Resistive Pull Up
Digital Output pin	Disconnect_LED Advertising_LED Connected_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.	**[Basic tab]** DeepSleepCapable = true
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	**[Basic 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 Running Speed and Cadence Server in the GAP Peripheral role. Also, the Battery and Device Information Services are included.

Figure 15. General Settings


Configure 'BLE'

Name: BLE

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

Load configuration | Save configuration

(●) 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)

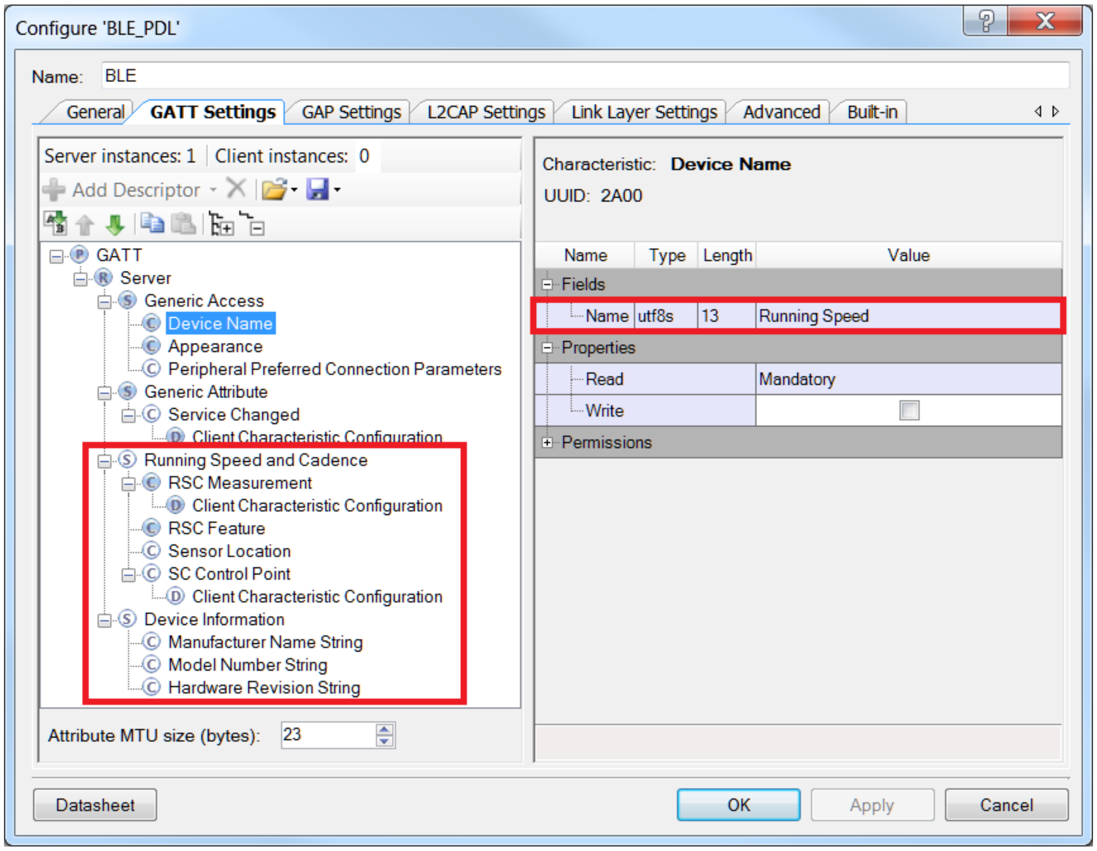
Over-The-Air bootloading with code sharing

(●) Disabled
( ) Stack and Profile
( ) Profile only

( ) BLE Controller only (HCI over UART)

Datasheet | OK | Apply | Cancel

Figure 16. 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
    - Running Speed and Cadence
      - RSC Measurement
        - Client Characteristic Configuration
      - RSC Feature
      - Sensor Location
      - SC Control Point
        - Client Characteristic Configuration
    - Device Information
      - Manufacturer Name String
      - Model Number String
      - Hardware Revision String

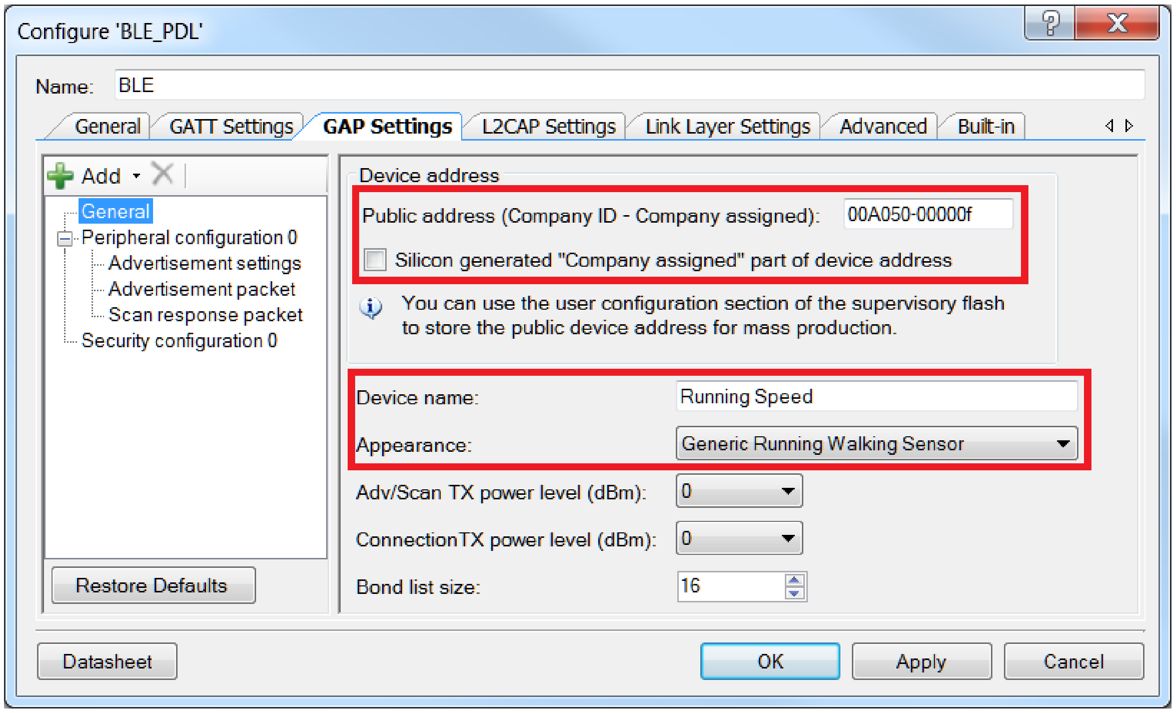
Characteristic: **Device Name**
UUID: 2A00

Name	Type	Length	Value
Fields			
Name	utf8s	13	Running Speed
Properties			
Read			Mandatory
Write			☐
Permissions			

Attribute MTU size (bytes): 23

Datasheet | OK | Apply | Cancel

Figure 17. GAP Settings



Name: BLE

Device address

Public address (Company ID - Company assigned): 00A050-00000f

☐ 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: Running Speed

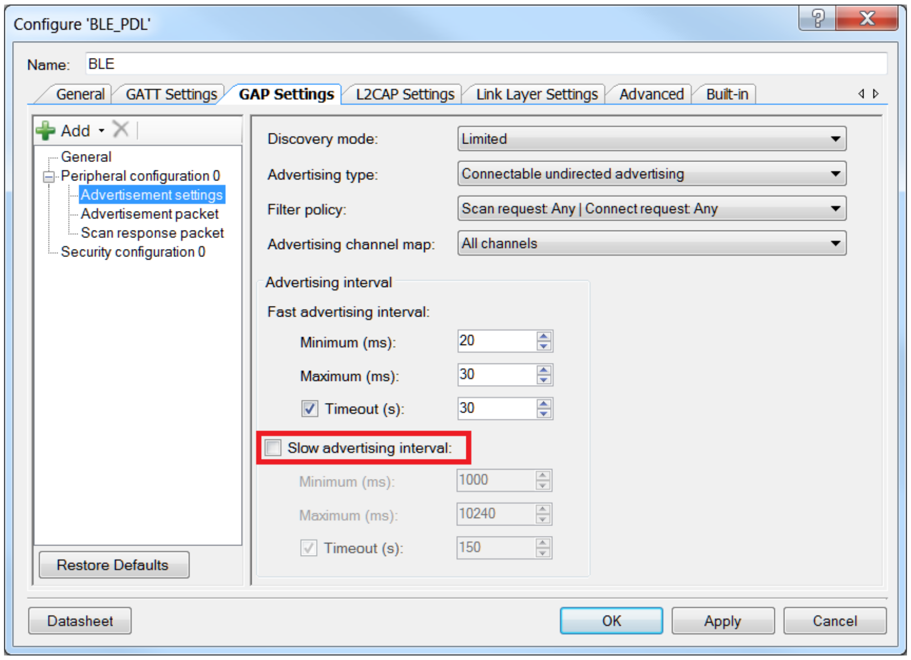
Appearance: Generic Running Walking Sensor

Adv/Scan TX power level (dBm): 0

ConnectionTX power level (dBm): 0

Bond list size: 16

Figure 18. GAP Settings: Advertisement Settings



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

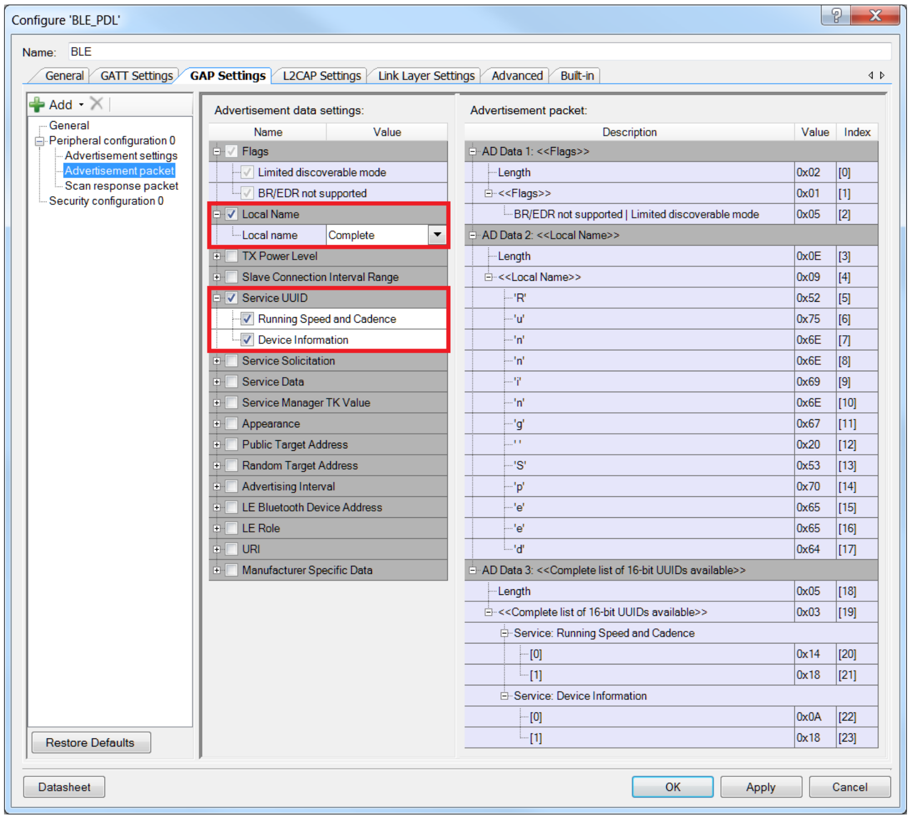
☐ Slow advertising interval:

Minimum (ms): 1000

Maximum (ms): 10240

✓ Timeout (s): 150

Figure 19. GAP Settings: Advertisement Packet



Configure 'BLE_PDL'

Name: BLE

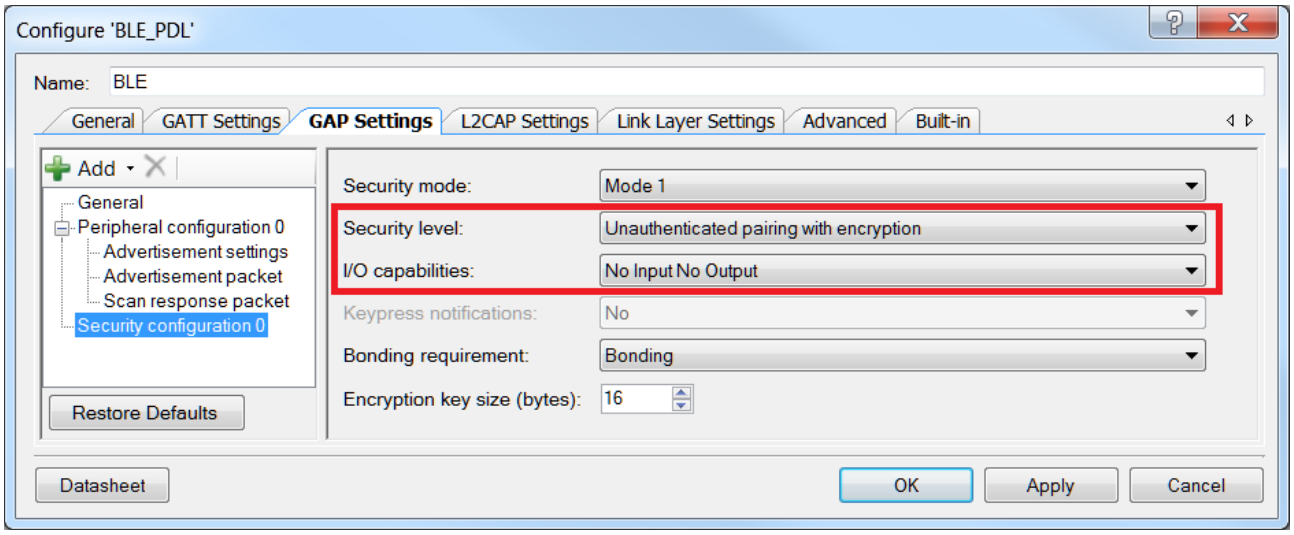
Advertisement data settings:

Name	Value
✓ Flags	
✓ Limited discoverable mode	
✓ BR/EDR not supported	
✓ Local Name	
Local name	Complete
TX Power Level	
Slave Connection Interval Range	
✓ Service UUID	
✓ Running Speed and Cadence	
✓ 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	0x0E	[3]
<<Local Name>>	0x09	[4]
'R'	0x52	[5]
'u'	0x75	[6]
'n'	0x6E	[7]
'n'	0x6E	[8]
'i'	0x69	[9]
'n'	0x6E	[10]
'g'	0x67	[11]
' '	0x20	[12]
'S'	0x53	[13]
'p'	0x70	[14]
'e'	0x65	[15]
'e'	0x65	[16]
'd'	0x64	[17]
AD Data 3: <<Complete list of 16-bit UUIDs available>>		
Length	0x05	[18]
<<Complete list of 16-bit UUIDs available>>	0x03	[19]
Service: Running Speed and Cadence		
[0]	0x14	[20]
[1]	0x18	[21]
Service: Device Information		
[0]	0x0A	[22]
[1]	0x18	[23]

Figure 20. Security Settings



Configure 'BLE_PDL'

Name: BLE

Security mode:	Mode 1
Security level:	Unauthenticated pairing with encryption
I/O capabilities:	No Input No Output
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.

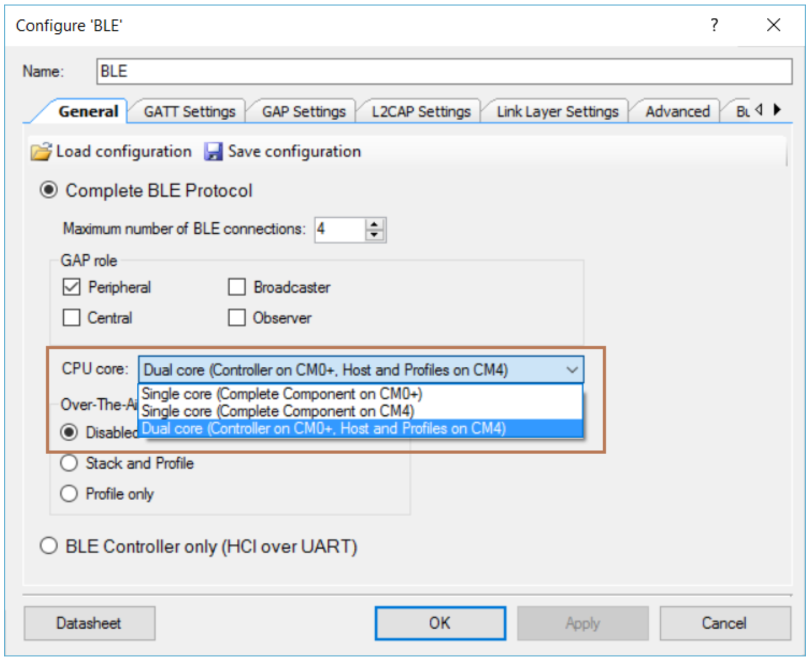
The BLE example structure allows easy switching between different CPU cores options. Important to remember:

- 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 21. Select CPU Core

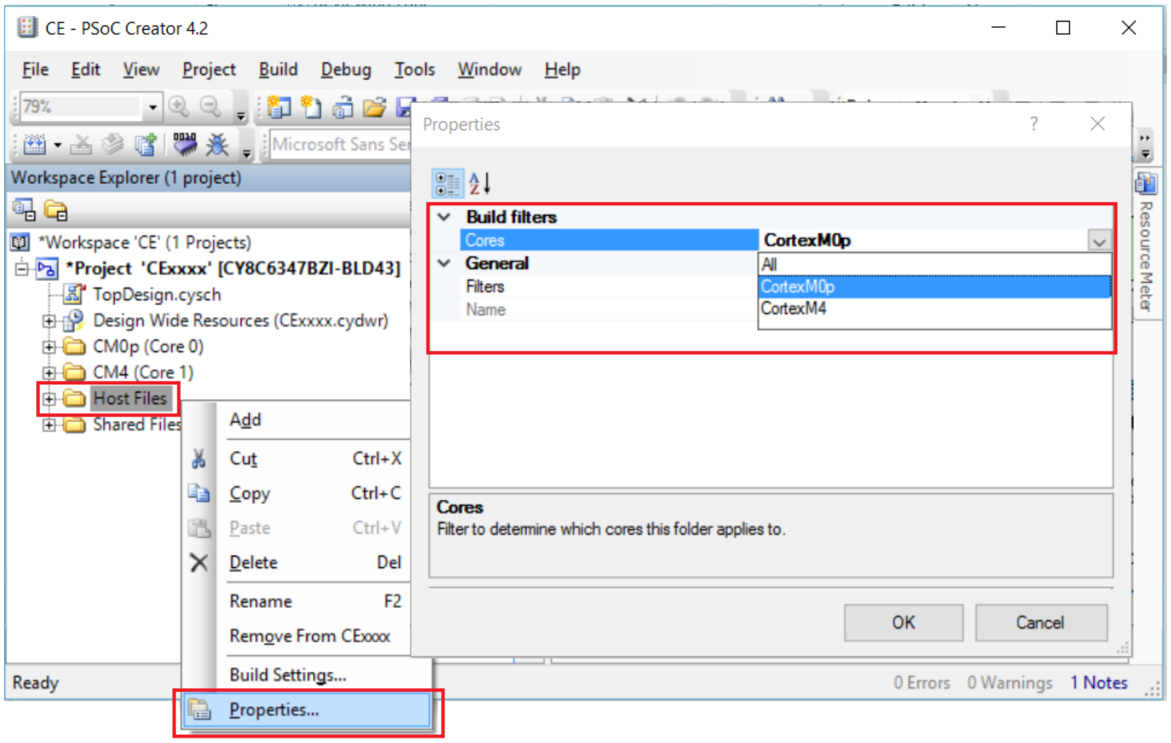


Complete BLE Protocol
Maximum number of BLE connections: 4
GAP role
Peripheral [✓]
Broadcaster
Central
Observer
CPU core: Dual core (Controller on CM0+, Host and Profiles on CM4)
Single core (Complete Component on CM0+)
Single core (Complete Component on CM4)
Dual core (Controller on CM0+, Host and Profiles on CM4)
Over-The-Air
[✓ 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 22.

   - 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 22. Change Core Properties

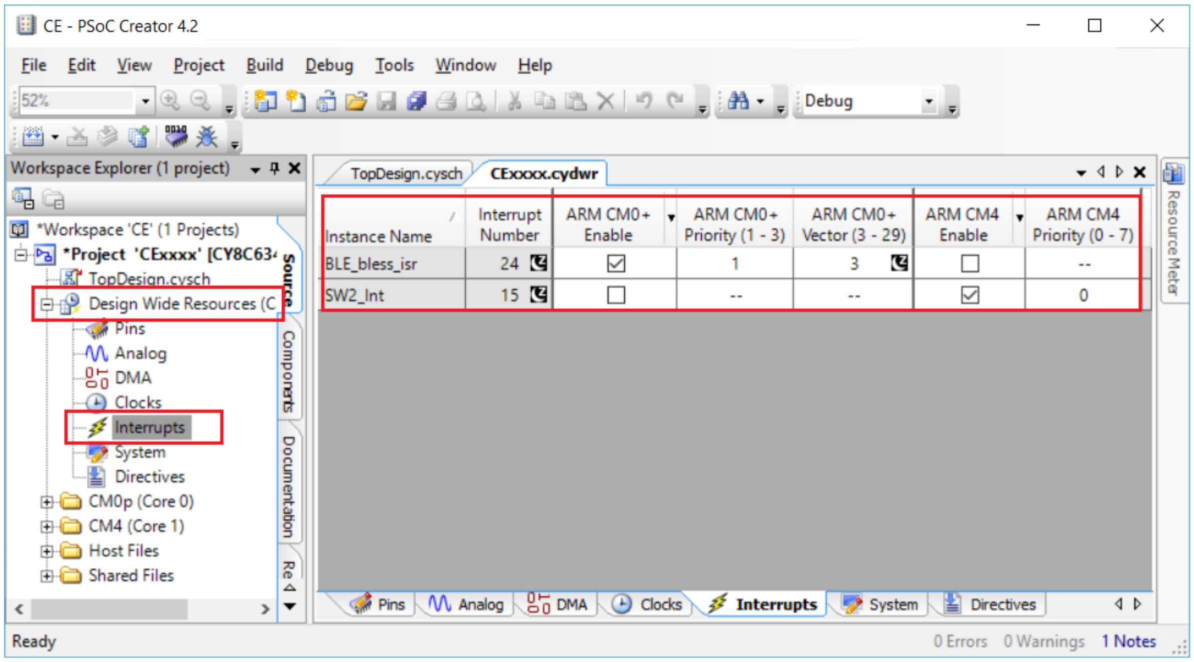


Cores
Filter to determine which cores this folder applies to.

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 23. Assign Interrupts


CE - PSoC Creator 4.2
File Edit View Project Build Debug Tools Window Help
Debug
Workspace Explorer (1 project)
*Workspace 'CE' (1 Projects)
*Project 'CExxxx' [CY8C63
TopDesign.cysch
Design Wide Resources (C
Pins
Analog
DMA
Clocks
Interrupts
System
Directives
CM0p (Core 0)
CM4 (Core 1)
Host Files
Shared Files
TopDesign.cysch
CExxxx.cydwr
Pins Analog DMA Clocks Interrupts System Directives
Ready
0 Errors 0 Warnings 1 Notes

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: CE217642 - BLE Running Speed and Cadence Profile with PSoC 6 MCU with BLE Connectivity

Document Number: 002-17642

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6091518	NPAL	05/14/2018	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

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