CE217639 - BLE Heart Rate with PSoC 6 MCU with BLE Connectivity

Objective

This example demonstrates the Heart Rate Client and Server operation of the Bluetooth Low Energy (BLE) PSoC Creator™ Component.

Overview

The Heart Rate Server and Heart Rate Client projects are used in a pair to demonstrate the Heart Rate Service (HRS) operation. The Heart Rate Server project demonstrates the BLE workflow procedures such as advertising, connecting, simulating, and notifying Heart Rate data and Battery Level. To conserve power, the device switches to Deep Sleep mode between the BLE connection intervals.

The Heart Rate Client project receives Heart Rate data from any BLE-enabled Heart Rate Sensor and indicates that data on any terminal software via a UART.

Requirements

Tool: PSoC Creator 4.2 or later

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

Associated Parts: All PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity (PSoC 6 BLE) parts

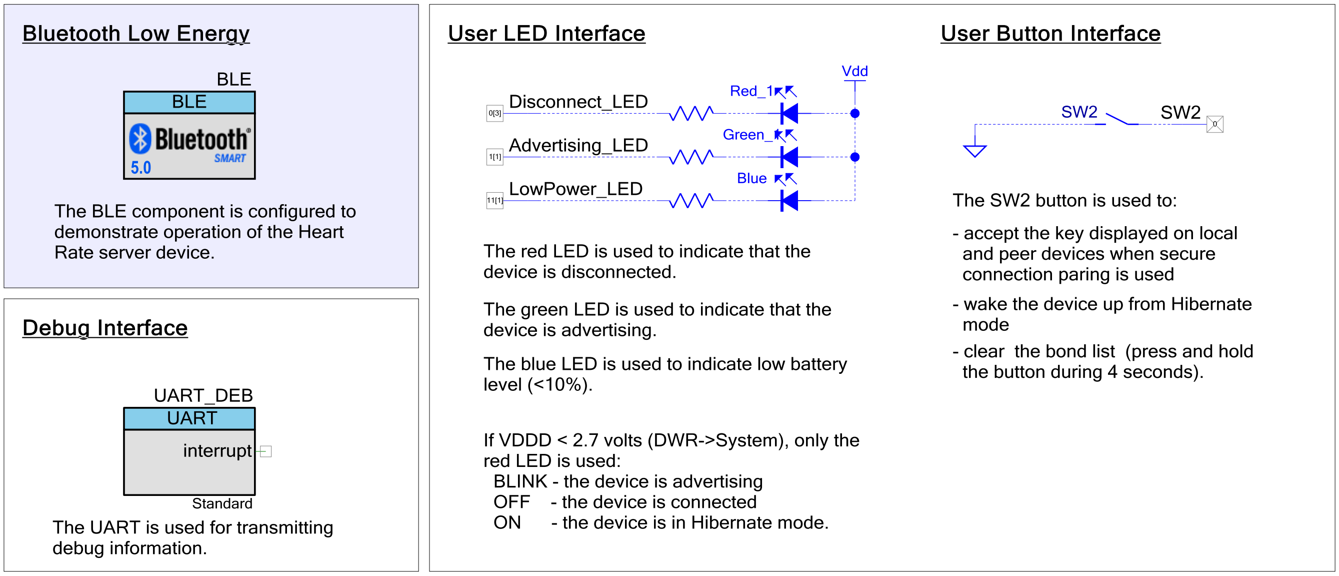
Related Hardware: CY8CKIT-062 PSoC 6 BLE Pioneer Kit

Heart Rate Server Code Example

Design

Figure 1 shows the top design schematic.

Figure 1. BLE Heart Rate Server Code-Example Schematic



The BLE component is configured to demonstrate operation of the Heart Rate server 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 low battery level (<10%).

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:

- accept the key displayed on local and peer devices when secure connection paring is used
- wake the device up from Hibernate mode
- clear the bond list (press and hold the button during 4 seconds).

The project automatically (not manually because the project is designed) starts the BLE Stack, and then starts the advertising GAP procedure (when `CY_BLE_STACK_ON` event is received). The green LED blinks while the device is advertising. After a connection request is received, the device performs the connection procedure and provides its GATT database (configured in the **GATT** tab) for a discovery process performed by the client. The supported services are: Generic Access (GAP) and Attribute (GATT) Services, Heart Rate Service (HRS), Battery Service (BAS), and Device Information Service (DIS). When the Heart Rate notification is enabled by the client, the project starts simulating all HRS related data (Heart Rate itself, Energy expended, RR-intervals). When the Battery Level notification is enabled by the client, the project starts to simulate and notify the Battery Level. The BLE stack timer is used to time the simulations, measurements, and LED blinking. The blue LED turns ON when the battery level value is less than 10 percent. The red LED is turned ON after disconnecting to indicate that no client is connected to the device. On the disconnection event, the device immediately starts advertising. When the device connects successfully, the red and green LEDs are turned OFF.

After a 180-second timeout expires, if no Central device has been connected, the Heart Rate Sensor stops advertising. The red LED turns ON indicating the disconnection state and the system enters Hibernate mode. Press the mechanical button (**SW2**) on the CY8CKIT-062 PSoC 6 BLE Pioneer Kit to wake up the system and restart advertising.

Design Considerations

Using UART for Debugging

Download and install a serial port communication program. Freeware such as Bray’s Terminal and PuTTY are available on the web.

1. Connect the PC and kit with a USB cable.
2. Open the device manager program in your PC, find a COM port that the kit is connected to, and note the port number.
3. Open the serial port communication program and select the previously noted COM port.
4. Configure the Baud rate, Parity, Stop bits, and Flow control information in the PuTTY configuration window. The default settings: 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.
5. Start communicating with the device as explained in the Operation section.

The UART debugging can be disabled by setting the `DEBUG_UART_ENABLED to DISABLED` in the *common.h* file.

Switching the CPU Cores Usage

This section describes how to switch between different CPU cores usage (Single core and Dual core) in the 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+ 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 examples’ structure allows easy switching between different CPU cores options.

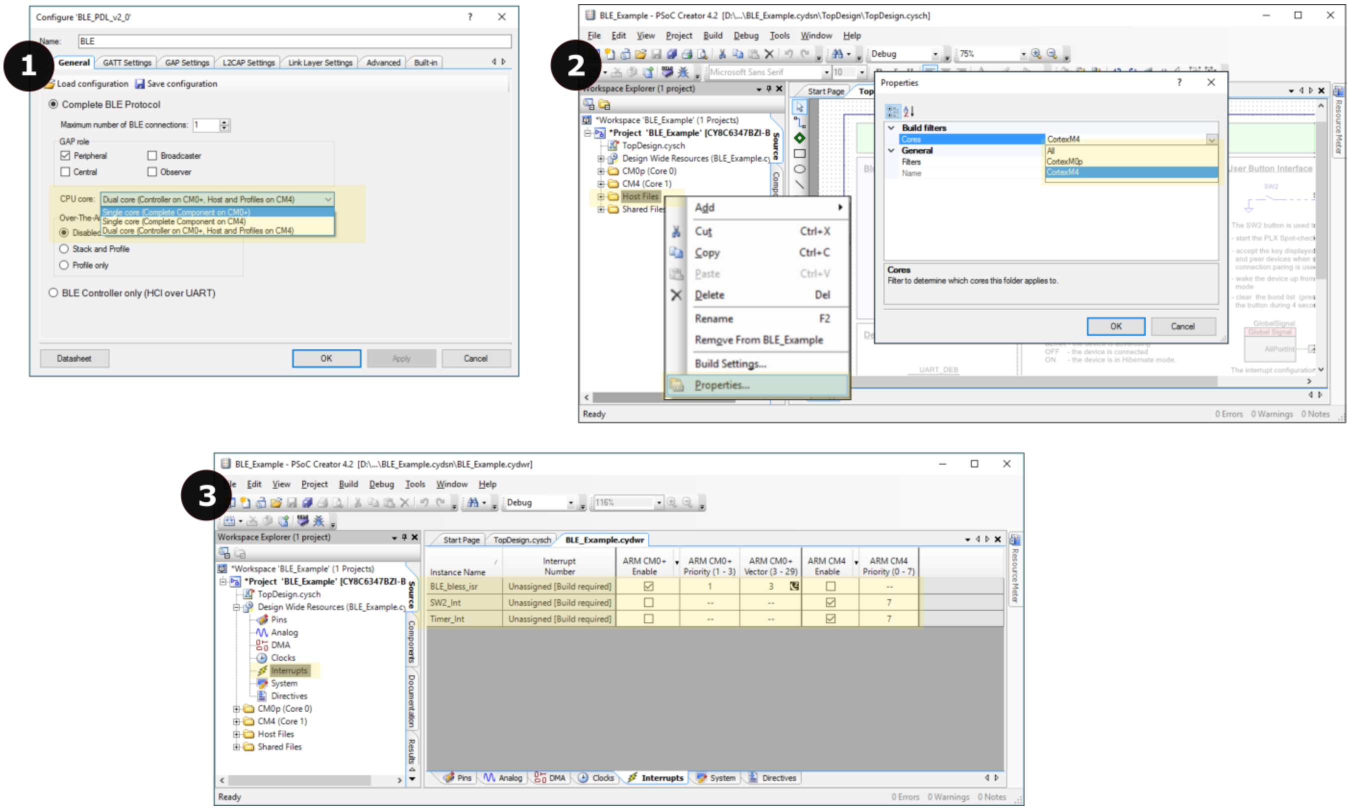
Important to remember:

- 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.
- All additional interrupts (SW2, MCWDT, and so on) used in the example must be assigned to the host core.

Do the following to switch the CPU cores usage:

1. In the BLE Component Customizer **General** tab, select the appropriate CPU core option.
2. Change the core properties to CortexM4 or CortexC0p for the project folder Host Files based on the CPU core option selected in step 1. It should be:
   - 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**
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 2. Steps for Switching the CPU Cores Usage



1

Configure 'BLE_PDL_v2_0'

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)

- 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

BLE_Example - PSoC Creator 4.2 [D:\...\BLE_Example.cydsn\TopDesign\TopDesign.cysch]

*Workspace 'BLE_Example' (1 Projects)

*Project 'BLE_Example' [CY8C6347BZI-B

TopDesign.cysch

Design Wide Resources (BLE_Example.cy

CM0p (Core 0)

CM4 (Core 1)

Host Files

Shared Files

Properties

Build filters

Cores	CortexM4

All
CortexM0p
CortexM4

General

Filters

Name

Cores

Filter to determine which cores this folder applies to.

3

BLE_Example - PSoC Creator 4.2 [D:\...\BLE_Example.cydsn\BLE_Example.cydwr]

*Workspace 'BLE_Example' (1 Projects)

*Project 'BLE_Example' [CY8C6347BZI-B

TopDesign.cysch

Design Wide Resources (BLE_Example.cy

Pins

Analog

DMA

Clocks

Interrupts

System

Directives

CM0p (Core 0)

CM4 (Core 1)

Host Files

Shared Files

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	Unassigned [Build required]	☑	1	3	☐	--
SW2_Int	Unassigned [Build required]	☐	--	--	☑	7
Timer_Int	Unassigned [Build required]	☐	--	--	☑	7

Hardware Setup

The code example was designed for the CY8CKIT-062 PSoC 6 BLE Pioneer Kit.

Table 1 lists the pin assignments and connections required on the development board for supported kits.

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]	
Advertising_LED	P1[1]	The green color of the RGB LED
Disconnect_LED	P0[3]	The red color of the RGB LED
LowPower_LED	P11[1]	The blue color of the RGB LED
SW2	P0[4]	

LED Behavior for V_{DDD} Voltage < 2.7 V

If the V_{DDD} voltage is set to less than 2.7 V in the DWR settings of the **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.

Components

Table 2 lists the PSoC Creator Components used in this example and the hardware resources used by each of the components.

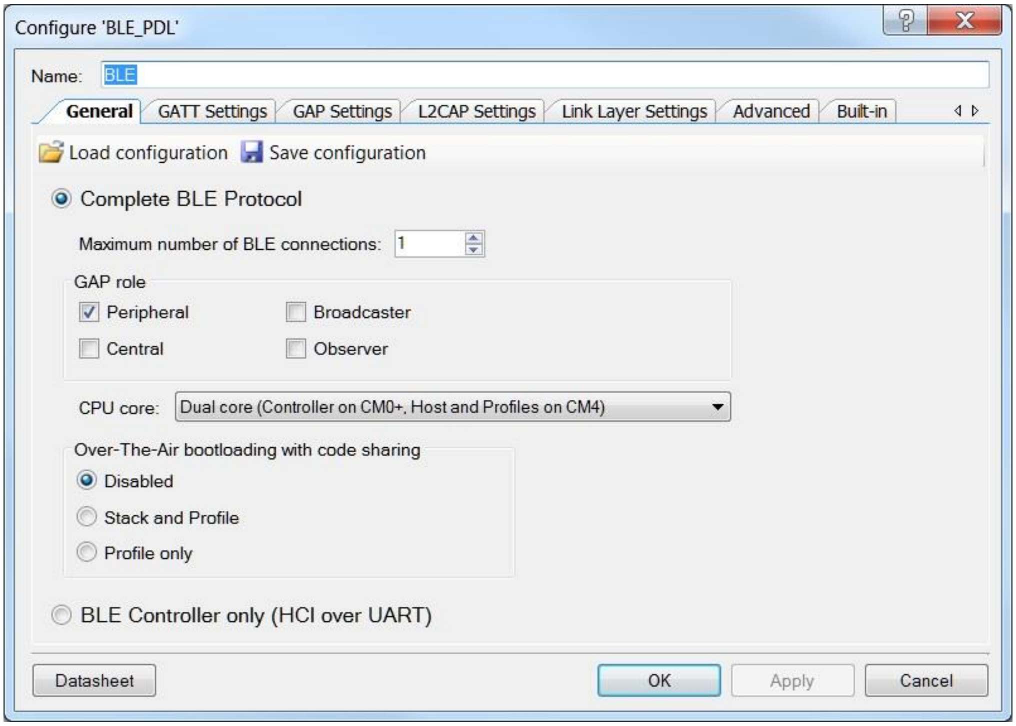
Table 2. PSoC Creator Components List

Component	Hardware Resources
UART_DEB	1 SCB
BLE	1 BLE, 1 Interrupt
SW2	1 pin
Disconnect_LED, Advertising_LED, LowPower_LED	3 pins

Parameter Settings

The BLE Component is configured as HRS Server in the GAP Peripheral role with the settings shown in Figure 3 to Figure 8.

Figure 3. General Settings



Configure 'BLE_PDL'

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

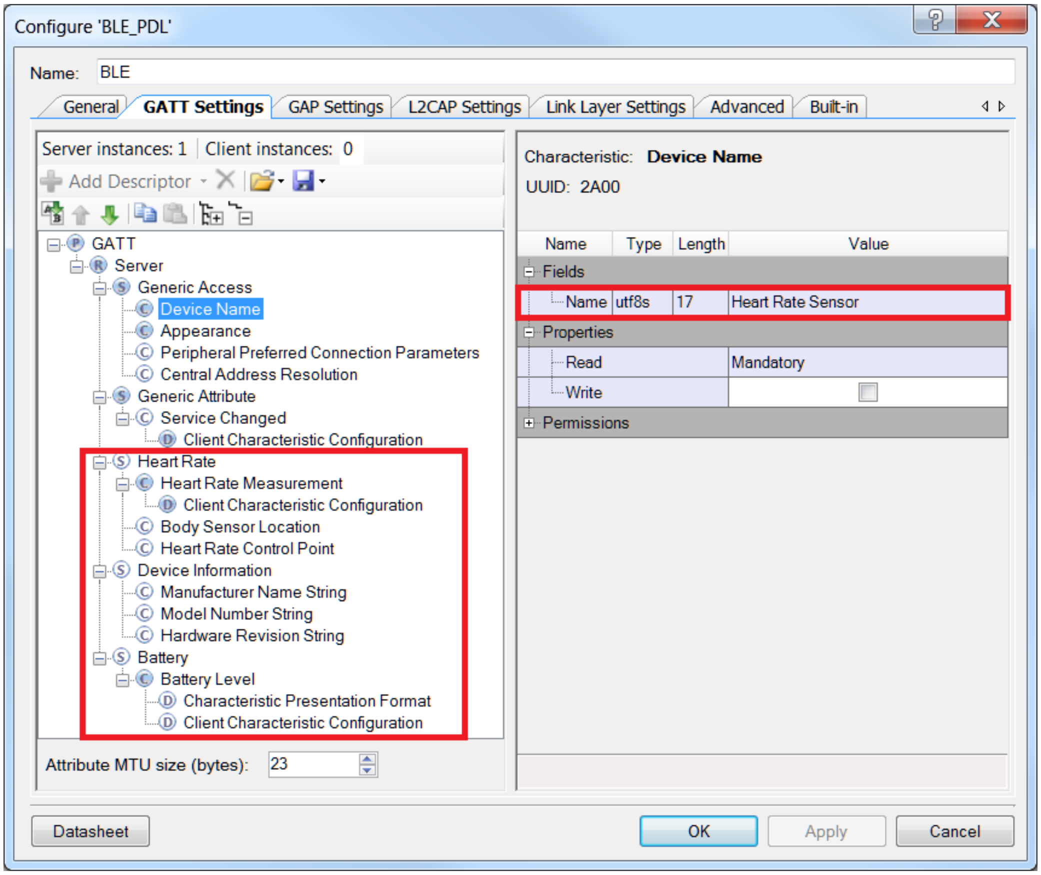
● Disabled

○ Stack and Profile

○ Profile only

○ BLE Controller only (HCI over UART)

Figure 4. GATT Settings



Name: BLE

Server instances: 1 | Client instances: 0

```
GATT
  Server
    Generic Access
      Device Name
      Appearance
      Peripheral Preferred Connection Parameters
      Central Address Resolution
    Generic Attribute
      Service Changed
        Client Characteristic Configuration
    Heart Rate
      Heart Rate Measurement
        Client Characteristic Configuration
      Body Sensor Location
      Heart Rate Control Point
    Device Information
      Manufacturer Name String
      Model Number String
      Hardware Revision String
    Battery
      Battery Level
        Characteristic Presentation Format
        Client Characteristic Configuration
```

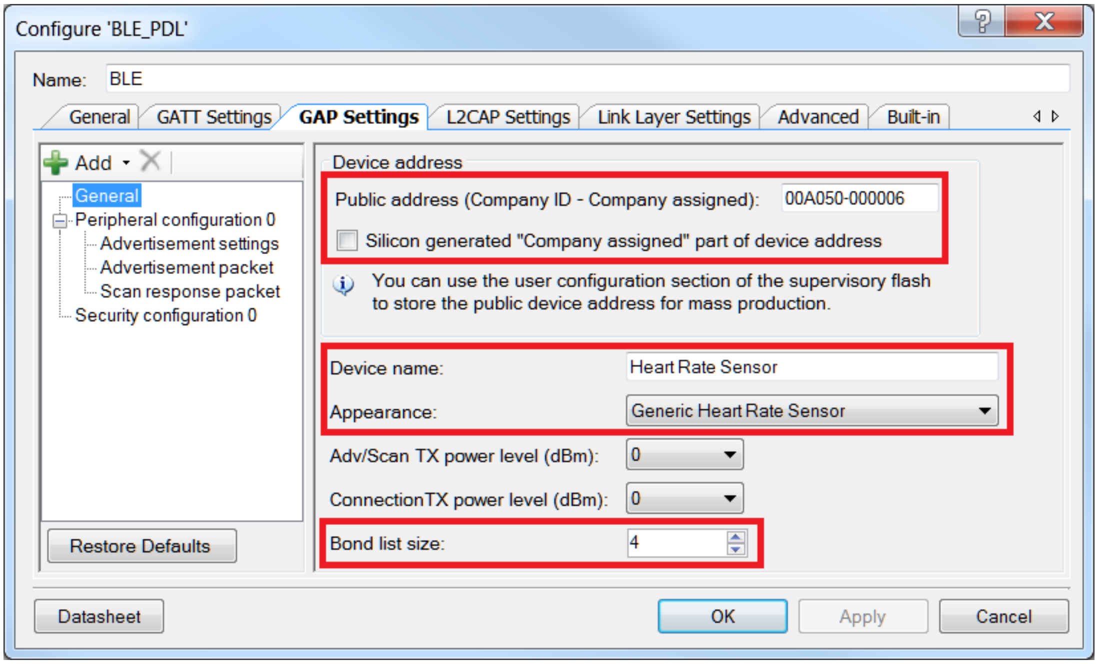
Attribute MTU size (bytes): 23

Characteristic: **Device Name**

UUID: 2A00

Name	Type	Length	Value
Fields			
Name	utf8s	17	Heart Rate Sensor
Properties			
Read			Mandatory
Write			□
Permissions			

Figure 5. GAP Settings



Name: BLE

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

Device address

Public address (Company ID - Company assigned): 00A050-000006

□ 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: Heart Rate Sensor

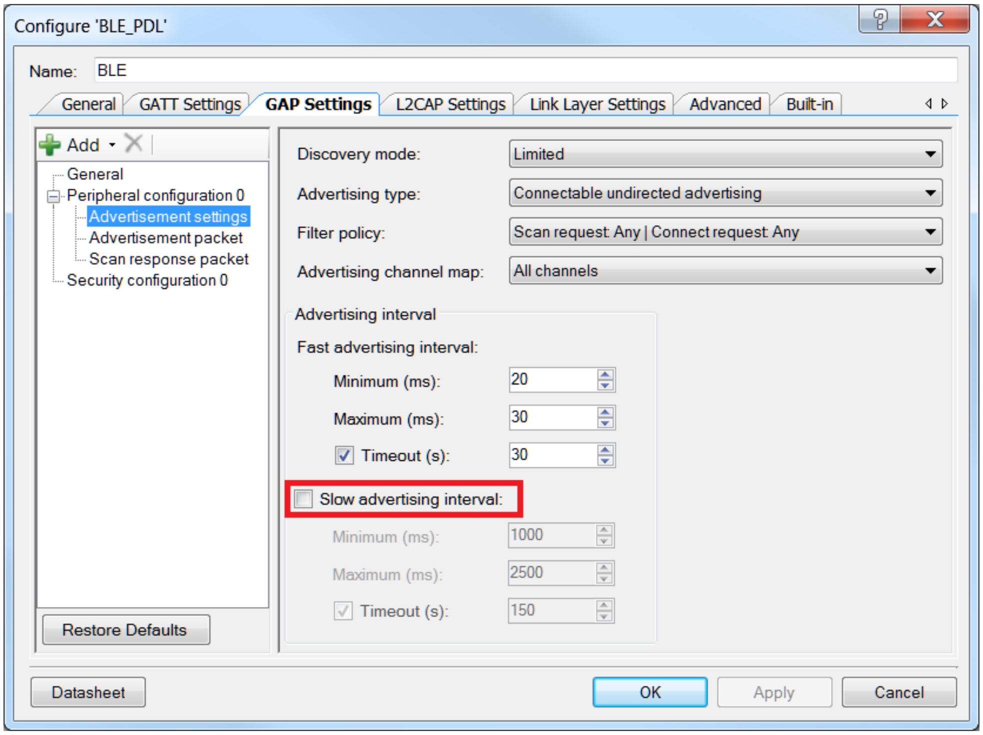
Appearance: Generic Heart Rate Sensor

Adv/Scan TX power level (dBm): 0

ConnectionTX power level (dBm): 0

Bond list size: 4

Figure 6. GAP Settings > Advertisement Setting



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

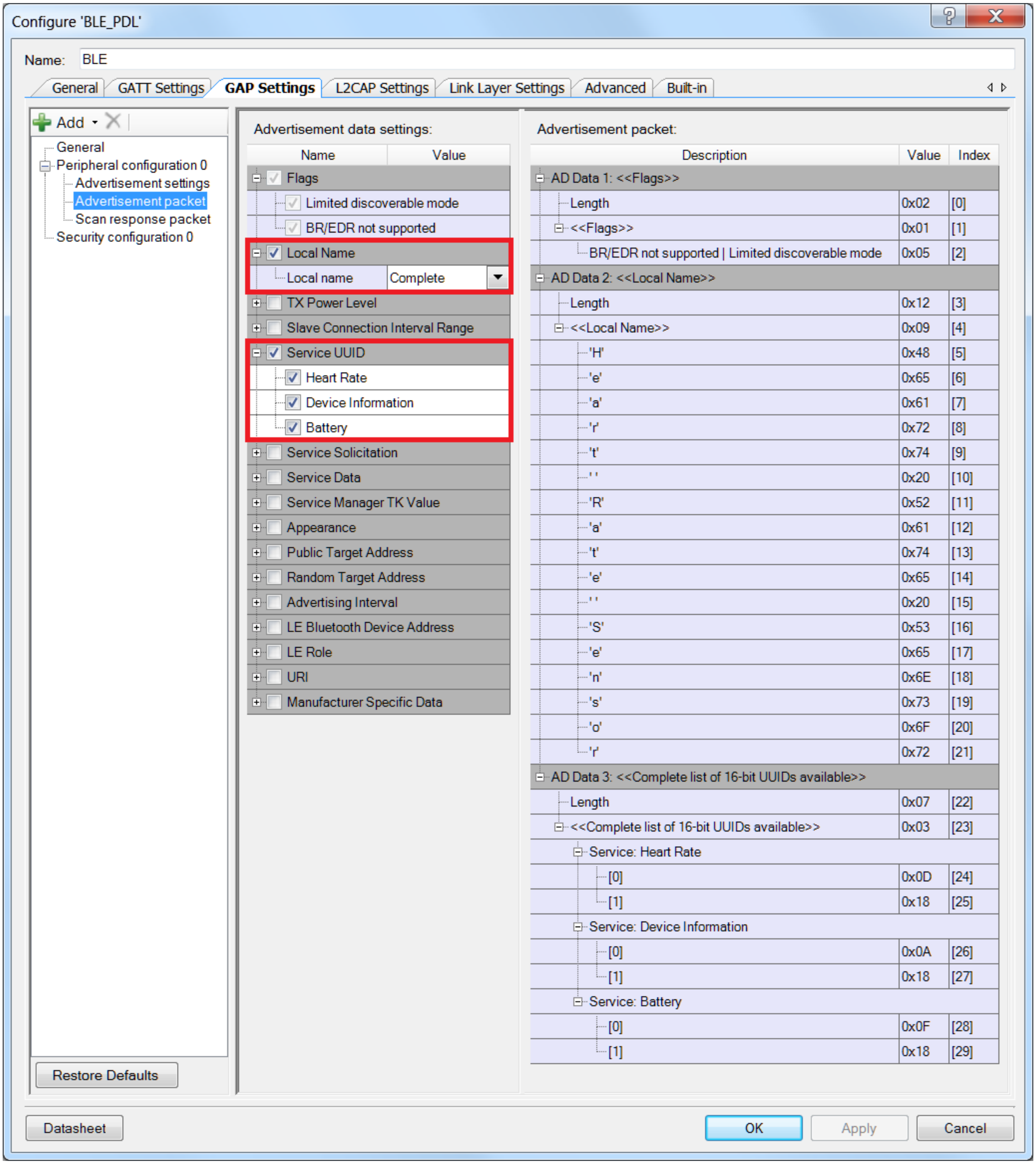
□ Slow advertising interval:

Minimum (ms): 1000

Maximum (ms): 2500

☑ Timeout (s): 150

Figure 7. GAP Settings > Advertisement Packet



Configure 'BLE_PDL'

Name: BLE

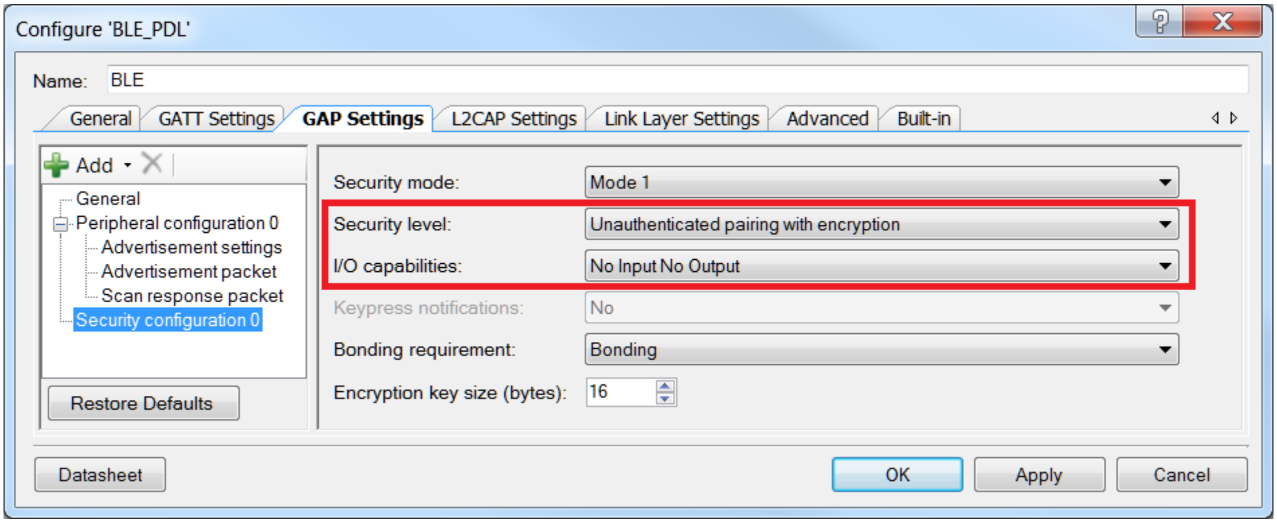
Advertisement data settings:

Name	Value
☑ Flags	
☑ Limited discoverable mode	
☑ BR/EDR not supported	
☑ Local Name	
Local name	Complete
□ TX Power Level	
□ Slave Connection Interval Range	
☑ Service UUID	
☑ Heart Rate	
☑ Device Information	
☑ Battery	
□ 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	0x12	[3]
<<Local Name>>	0x09	[4]
'H'	0x48	[5]
'e'	0x65	[6]
'a'	0x61	[7]
'r'	0x72	[8]
't'	0x74	[9]
' '	0x20	[10]
'R'	0x52	[11]
'a'	0x61	[12]
't'	0x74	[13]
'e'	0x65	[14]
' '	0x20	[15]
'S'	0x53	[16]
'e'	0x65	[17]
'n'	0x6E	[18]
's'	0x73	[19]
'o'	0x6F	[20]
'r'	0x72	[21]
AD Data 3: <<Complete list of 16-bit UUIDs available>>		
Length	0x07	[22]
<<Complete list of 16-bit UUIDs available>>	0x03	[23]
Service: Heart Rate		
[0]	0x0D	[24]
[1]	0x18	[25]
Service: Device Information		
[0]	0x0A	[26]
[1]	0x18	[27]
Service: Battery		
[0]	0x0F	[28]
[1]	0x18	[29]

Figure 8. GAP Settings > Security Configuration



Security mode: Mode 1

Security level: Unauthenticated pairing with encryption

I/O capabilities: No Input No Output

Keypress notifications: No

Bonding requirement: Bonding

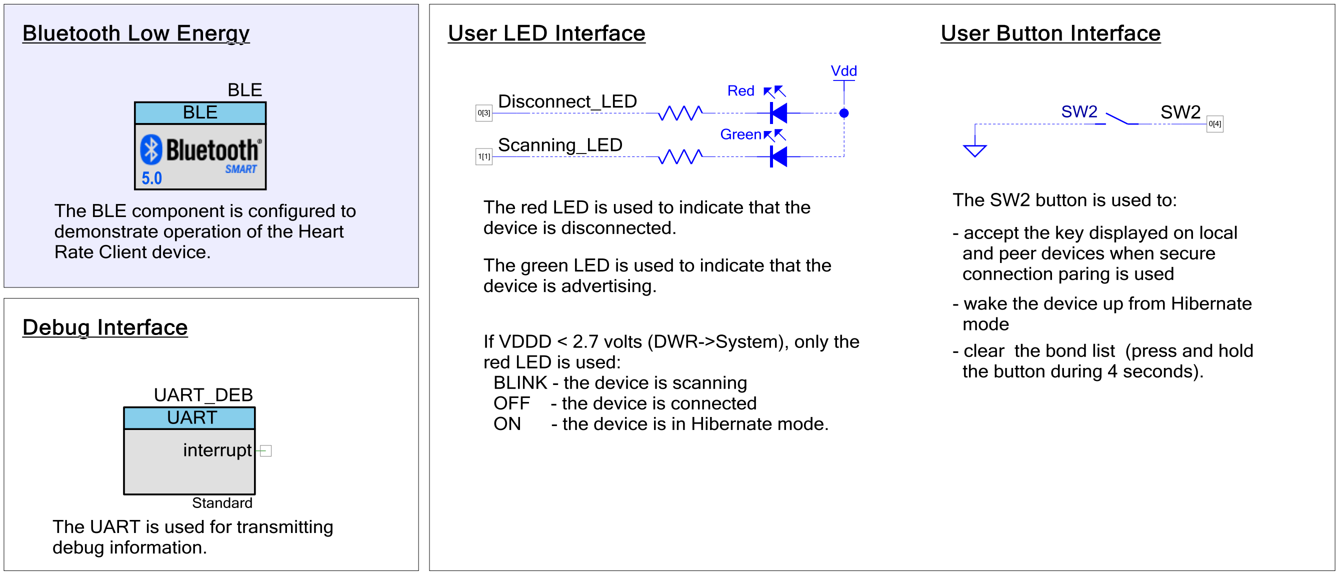
Encryption key size (bytes): 16

Heart-Rate Client Code Example

Design

Figure 9 shows the top design schematic.

Figure 9. Heart-Rate Client Code Example Schematic



The BLE component is configured to demonstrate operation of the Heart Rate Client 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 scanning
OFF - the device is connected
ON - the device is in Hibernate mode.

The SW2 button is used to:

- accept the key displayed on local and peer devices when secure connection paring is used
- wake the device up from Hibernate mode
- clear the bond list (press and hold the button during 4 seconds).

The project demonstrates the BLE workflow procedures, such as scanning, discovering, connecting, writing/reading characteristics/descriptors, and receiving notifications. It is designed to work as a pair with the BLE Heart-Rate Sensor Example Project.

The working of the project includes:

- Automatically (not manually because the project is designed) starting the BLE Stack, and then starting the scanning GAP procedure (when `STACK_ON` event is received)
- Receiving and parsing advertisement data, and the green LED blinks while the device is scanning
- Finding the HRS UUID in the advertisement packet and immediately connecting to that device, and discovering all supported primary services (configured in the **GATT** tab). In this example, they are the Generic Access (GAP) and Attribute (GATT) Services, then the Heart Rate (HRS), Battery (BAS), and Device Information Services (DIS).
- Discovering included services (which may be secondary) and characteristics of each above mentioned primary services.
- Discovering descriptors of each service characteristic which can have descriptors.
- After the discovery process (when the `DISCOVERY_COMPLETE` event is received), sending a request to read the Body Sensor Location characteristic and waiting for the `HRSC_BSL_READ_RESPONSE` event in the Heart-Rate Profile's callback (`HeartRateCallBack()`).
- On this event, indicating the received Body Sensor Location value and enabling the Heart Rate Measurement Notification. The notifications come approximately once a second.
- Enabling the Battery Level notification, which comes immediately after enabling and then when the battery level changes.

The red LED is turned ON after disconnecting to indicate that no server is connected to the device. On the disconnection event, the device immediately starts scanning peripherals. When the Central device connects successfully, the red and green LEDs are turned OFF.

After a 180-second timeout, if no peripheral device has been connected, the Heart Rate Collector stops discovering, the red LED is turned ON indicating the disconnection state and the system enters Hibernate mode. Press the mechanical button **SW2** on the CY8CKIT-062 PSoC 6 BLE Pioneer Kit to wake up the system and start discovering.

Design Considerations

Using UART for Debugging

Download and install a serial port communication program. Freeware such as Bray's Terminal and PuTTY are available on the web.

1. Connect the PC and kit with a USB cable.
2. Open the device manager program in your PC, find a COM port that the kit is connected to, and note the port number.
3. Open the serial port communication program and select the previously noted COM port.
4. Configure the Baud rate, Parity, Stop bits, and Flow control information in the PuTTY configuration window. The default settings: 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.
5. Start communicating with the device as explained in the Operation section.

The UART debugging can be disabled by setting the `DEBUG_UART_ENABLED` to `DISABLED` in the *common.h* file.

Hardware Setup

The code example was designed for the CY8CKIT-062 PSoC 6 BLE Pioneer Kit.

Table 3 lists the pin assignments and connections required on the development board for supported kits.

Table 3. 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]	
Scanning_LED	P1[1]	The green color of the RGB LED.
Disconnect_LED	P0[3]	The red color of the RGB LED.
SW2	P0[4]	

LED Behavior for V_{DDD} Voltage < 2.7 V

If the V_{DDD} voltage is set to less than 2.7 V in the DWR settings of the **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.

Components

Table 4 lists the PSoC Creator Components used in this example and the hardware resources used by each of the components.

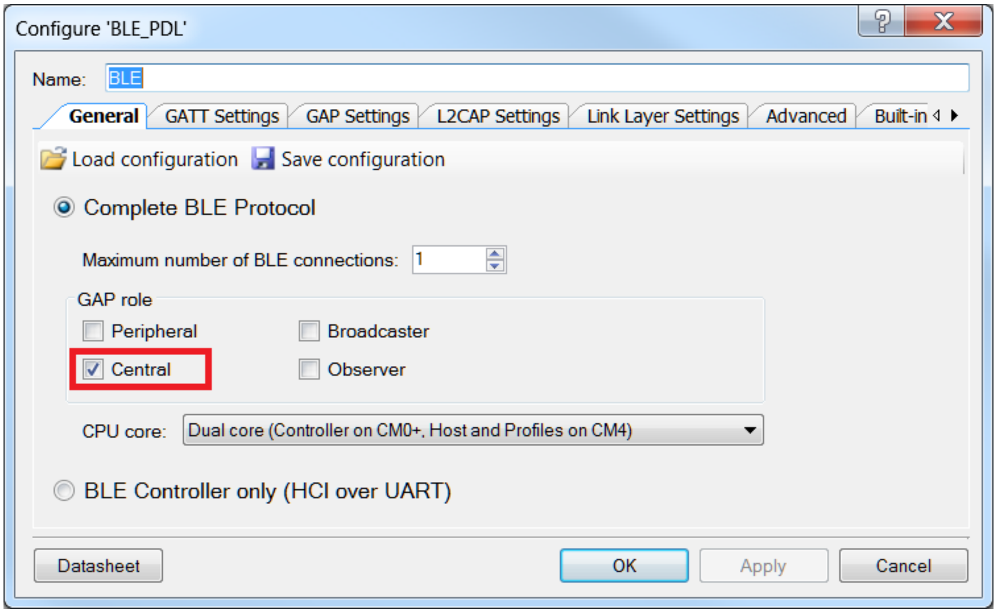
Table 4. PSoC Creator Components List

Component	Hardware Resources
UART_DEB	1 SCB
BLE	1 BLE, 1 Interrupt
SW2	1 pin
Disconnect_LED Scanning_LED	2 pins

Parameter Settings

The BLE Component is configured as an HPS Client in the GAP Central role with the settings shown in Figure 10 to Figure 14.

Figure 10. General Settings



Configure 'BLE_PDL'

Name: BLE

● Complete BLE Protocol

Maximum number of BLE connections: 1

GAP role

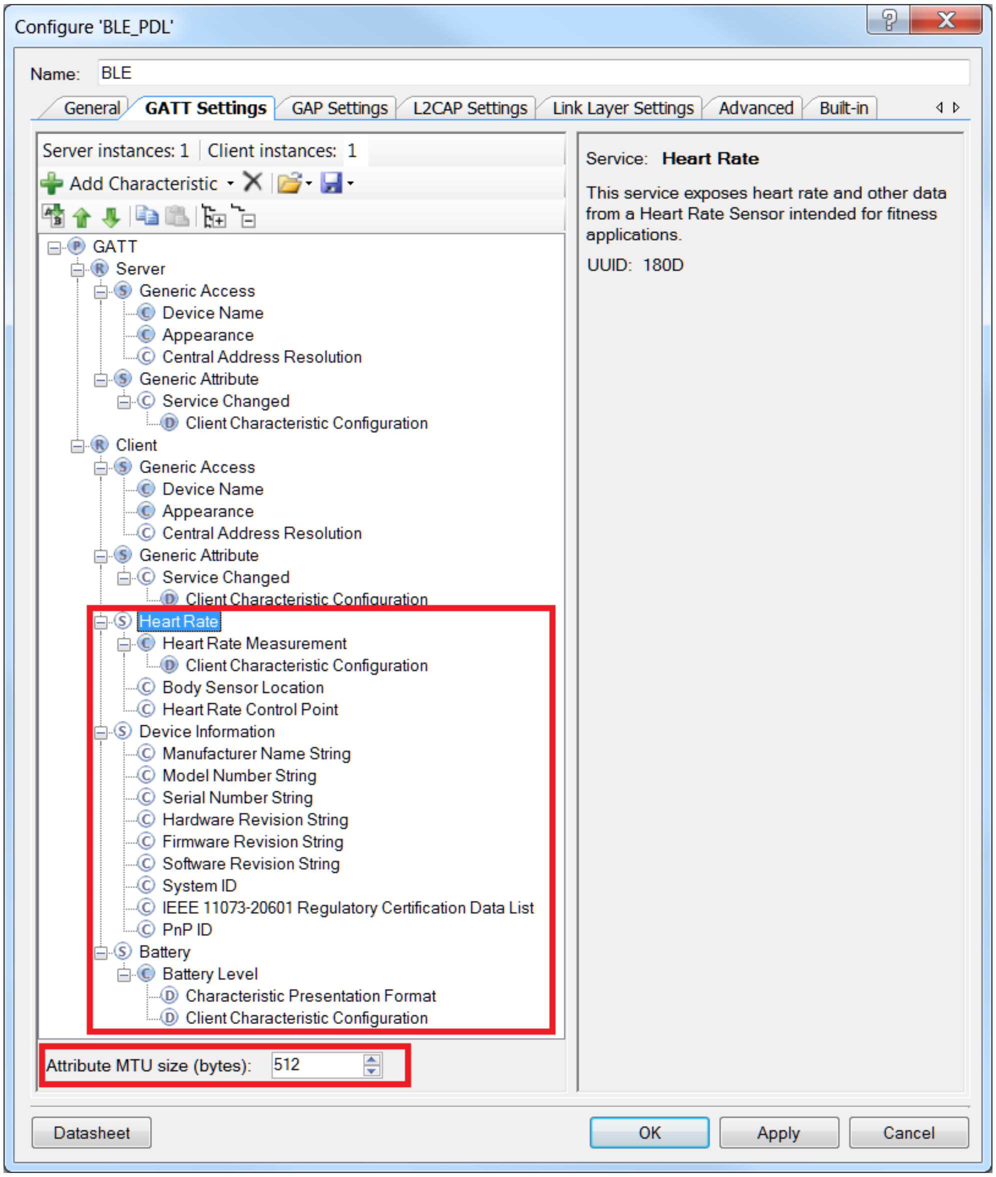
□ Peripheral □ Broadcaster

☑ Central □ Observer

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

○ BLE Controller only (HCI over UART)

Figure 11. GATT Settings



Name: BLE

Server instances: 1 | Client instances: 1

- GATT
  - Server
    - Generic Access
      - Device Name
      - Appearance
      - Central Address Resolution
    - Generic Attribute
      - Service Changed
        - Client Characteristic Configuration
  - Client
    - Generic Access
      - Device Name
      - Appearance
      - Central Address Resolution
    - Generic Attribute
      - Service Changed
        - Client Characteristic Configuration
    - Heart Rate
      - Heart Rate Measurement
        - Client Characteristic Configuration
      - Body Sensor Location
      - Heart Rate Control Point
    - 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
    - Battery
      - Battery Level
        - Characteristic Presentation Format
        - Client Characteristic Configuration

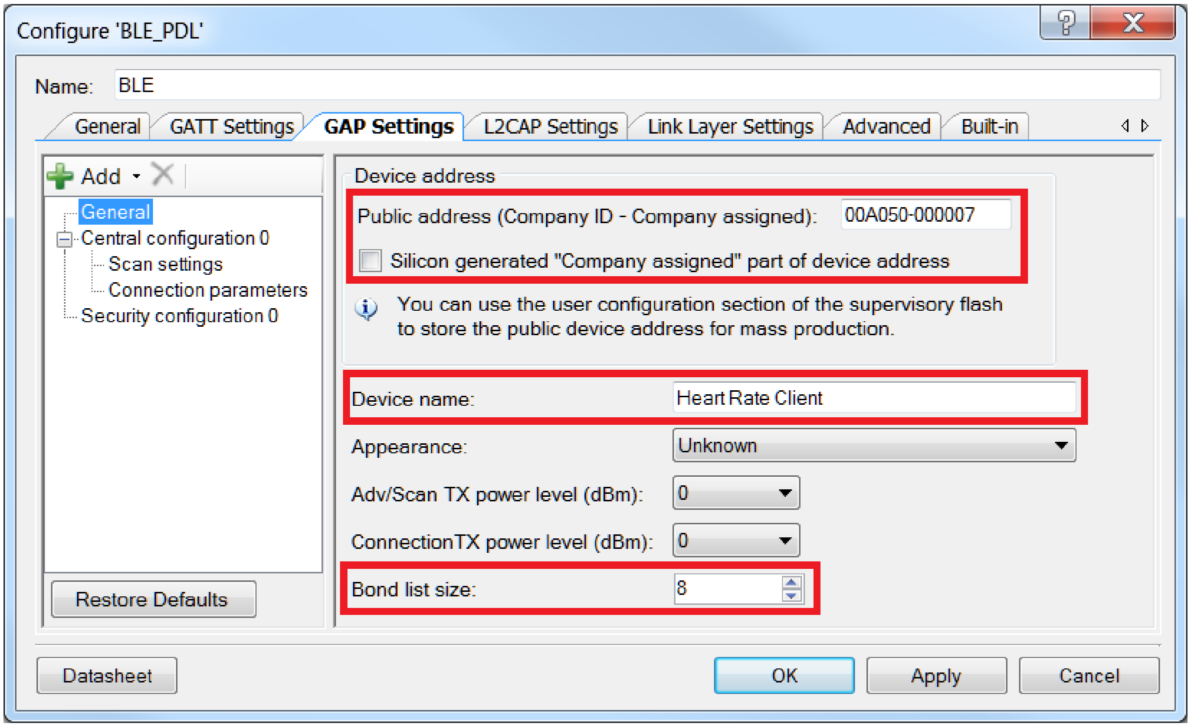
Attribute MTU size (bytes): 512

Service: **Heart Rate**

This service exposes heart rate and other data from a Heart Rate Sensor intended for fitness applications.

UUID: 180D

Figure 12. GAP Settings



Configure 'BLE_PDL'

Name: BLE

Device address

Public address (Company ID - Company assigned): 00A050-000007

□ 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: Heart Rate Client

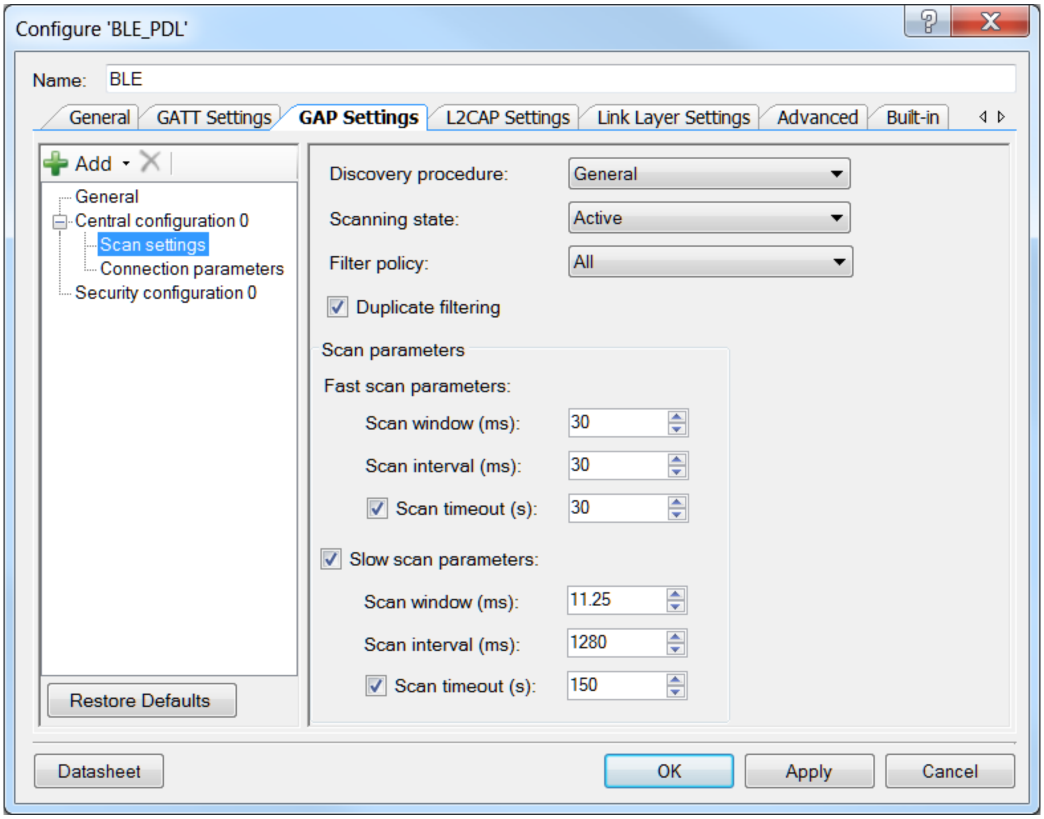
Appearance: Unknown

Adv/Scan TX power level (dBm): 0

ConnectionTX power level (dBm): 0

Bond list size: 8

Figure 13. GAP Settings > Scan Settings



Configure 'BLE_PDL'

Name: BLE

Discovery procedure: General

Scanning state: Active

Filter policy: All

☑ Duplicate filtering

Scan parameters

Fast scan parameters:

Scan window (ms): 30

Scan interval (ms): 30

☑ Scan timeout (s): 30

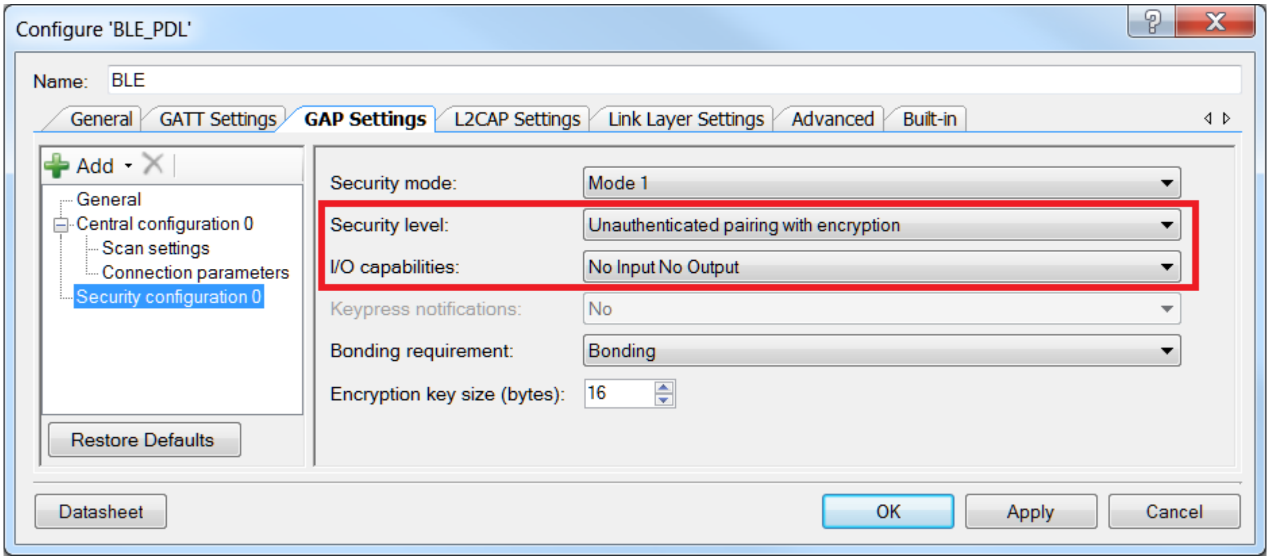
☑ Slow scan parameters:

Scan window (ms): 11.25

Scan interval (ms): 1280

☑ Scan timeout (s): 150

Figure 14. GAP Settings > Security Configuration



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

Operation

1. Build and program the BLE Heart Rate Service Server and BLE Heart Rate Service Client into the CY8CKIT-062 PSoC 6 BLE Pioneer Kit with PSoC 6 BLE.

2. Run two HyperTerminal (Bray's Terminal, PuTTY, and so on) instances: one for the BLE Heart Rate Service Server and another for the BLE Heart Rate Service Client.

3. In the **BLE Heart-Rate Service Client HyperTerminal** window, the client automatically starts scanning for advertising devices. When the scan report from the device with address **0x00a050000006** is received, press '**c**'. Then, select the number corresponding to the device with address **0x00a050000006**. An approximate example of an output on the client's HyperTerminal may appear as follows:

```
BLE Heart Rate Collector Example Project
BLE Stack Version: 5.0.0.718

CY_BLE_EVT_STACK_ON, StartAdvertisement
When you see a scan report from the device you wish to connect to - press 'c'.

CY_BLE_EVT_SET_DEVICE_ADDR_COMPLETE
CY_BLE_EVT_LE_SET_EVENT_MASK_COMPLETE
CY_BLE_EVT_GET_DEVICE_ADDR_COMPLETE: 00a050000007
CY_BLE_EVT_SET_TX_PWR_COMPLETE
CY_BLE_EVT_SET_TX_PWR_COMPLETE
CY_BLE_EVT_GAPC_SCAN_START_STOP
GAPC_START_SCANNING

ADV type: 0x0 address: 1123ac17c7c5, rssi - -90 dBm, data - 02 01 06 0e 09 43 55 53 54 4f
4d 20 53 65 72 76 65 72 03 19 00 00
ADV type: 0x0 address: 1123ac17c4ef, rssi - -70 dBm, data - 02 01 06 0e 09 43 79 63 6c 69
6e 67 20 50 6f 77 65 72 03 03 18 18 03 19 00 00
CY_BLE_EVT_GAP_KEYS_GEN_COMPLETE

-----------------------------------------------------------------------------
uuid: HEART RATE SERVICE - YES, added to the connect list
ADV type: 0x0 address: 00a050000006, rssi - -48 dBm, data - 02 01 05 12 09 48 65 61 72 74
20 52 61 74 65 20 53 65 6e 73 6f 72 07 03 0d 18 0a 18 0f 18
-----------------------------------------------------------------------------

ADV type: 0x0 address: 1123ac17c7a0, rssi - -87 dBm, data - 02 01 06 05 09 43 47 4d 53 03
02 1f 18 07 17 41 10 32 54 76 98
ADV type: 0x0 address: 00a050242001, rssi - -91 dBm, data - 02 01 02
ADV type: 0x0 address: 00a05060544e, rssi - -85 dBm, data - 02 01 06 0c 09 43 59 38 43 4b
49 54 2d 31 34 35

c
Detected device:
Device 1: 00a050000006
select device for connection:  (1..1):
1
Connecting to the device: 00a050000006
CY_BLE_EVT_GATT_CONNECT_IND: 3, b
CY_BLE_EVT_GAPC_SCAN_START_STOP
Scan complete!

CY_BLE_EVT_GAP_SMP_NEGOTIATED_AUTH_INFO: bdHandle=b, security=1, bonding=1, ekeySize=10,
err=0
CY_BLE_EVT_GAP_ENCRYPT_CHANGE: 0
CY_BLE_EVT_GAP_KEYINFO_EXCHNGE_CMPLT
CY_BLE_EVT_GAP_AUTH_COMPLETE: security:1, bonding:1, ekeySize:10, authErr 0

StartDiscovery
CY_BLE_EVT_STACK_BUSY_STATUS: 1
CY_BLE_EVT_PENDING_FLASH_WRITE
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_STACK_BUSY_STATUS: 0
Discovery complete.
Service with UUID 0x1800 has range from 0x1 to 0x9
Service with UUID 0x1801 has range from 0xa to 0xd
Service with UUID 0x180f has range from 0x1d to 0x21
Service with UUID 0x180a has range from 0x16 to 0x1c
Service with UUID 0x180d has range from 0xe to 0x15
Heart Rate Characteristic
CY_BLE_HRS_HRM   - Handle: 0x0010.
CY_BLE_HRS_BSL   - Handle: 0x0013.
CY_BLE_HRS_CPT   - Handle: 0x0015.

Heart Rate Measurement CCCD. Handle: 0x0011
Body Sensor Location Read Request is sent
Body Sensor Location: WRIST (2)
HRM CCCD Write Request is sent
Heart Rate Measurement Notification is Enabled
HRM CCCD Read Request is sent
HRM CCCD Read Response: 0001
 Battery Level CCCD Write Request is sent

Heart Rate Notification: Heart Rate: 108    EnergyExpended: 0    RR-Interval 0: 555
Heart Rate Notification: Heart Rate: 120    EnergyExpended: 0    RR-Interval 0: 500    RR-
Interval 1: 501
Heart Rate Notification: Sensor Contact is supported but not detected
Heart Rate Notification: Sensor Contact is supported but not detected
Heart Rate Notification: Sensor Contact is supported but not detected
Heart Rate Notification: Heart Rate: 168    EnergyExpended: 0    RR-Interval 0: 357    RR-
Interval 1: 358
Heart Rate Notification: Heart Rate: 180    EnergyExpended: 0    RR-Interval 0: 333    RR-
Interval 1: 334    RR-Interval 2: 335
Heart Rate Notification: Heart Rate: 192    EnergyExpended: 0    RR-Interval 0: 312    RR-
Interval 1: 313    RR-Interval 2: 314
Heart Rate Notification: Sensor Contact is supported but not detected
Heart Rate Notification: Sensor Contact is supported but not detected
Heart Rate Notification: Sensor Contact is supported but not detected
Heart Rate Notification: Heart Rate: 240    EnergyExpended: 0    RR-Interval 0: 250    RR-
Interval 1: 251    RR-Interval 2: 252    RR-Interval 3: 253
Heart Rate Notification: Heart Rate: 252    EnergyExpended: 0    RR-Interval 0: 238    RR-
Interval 1: 239    RR-Interval 2: 240    RR-Interval 3: 241
CY_BLE_EVT_TIMEOUT: 3
Heart Rate Notification: Heart Rate: 264    EnergyExpended: 0    RR-Interval 0: 227    RR-
Interval 1: 228    RR-Interval 2: 229    RR-Interval 3: 230
```

The HRS Server sends the Heart Rate and Battery Level notifications to the Central Client device. The output from the server's HyperTerminalt may appear as follows:

```
BLE Heart Rate Sensor Example Project
BLE Stack Version: 5.0.0.718

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: 00a050000006
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: 3, 7
CY_BLE_EVT_GAP_DEVICE_CONNECTED: connIntv = 28 ms
CY_BLE_EVT_GAP_AUTH_REQ: bdHandle=7, security=1, bonding=1, ekeySize=10, err=0
CY_BLE_EVT_GAP_SMP_NEGOTIATED_AUTH_INFO: bdHandle=7, security=1, bonding=1, ekeySize=10, err=0
CY_BLE_EVT_GAP_ENCRYPT_CHANGE: 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: 0, pending: 0
SimulBatteryLevelUpdate: 3
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ: handle: 13
Heart Rate Measurement Notification is Enabled
Store bonding data, status: 0, pending: 0
CY_BLE_EVT_GATTS_READ_CHAR_VAL_ACCESS_REQ: handle: 11
The Heart Rate Notification is sent successfully, Heart Rate = 96.
SimulBatteryLevelUpdate: 4
CY_BLE_EVT_GATTS_WRITE_REQ attr handle: 002, value: 01 00
The Heart Rate Notification is sent successfully, Heart Rate = 108.
SimulBatteryLevelUpdate: 5
The Heart Rate Notification is sent successfully, Heart Rate = 120.
The Heart Rate Notification is sent successfully, Heart Rate = 132.
SimulBatteryLevelUpdate: 6
The Heart Rate Notification is sent successfully, Heart Rate = 144.
SimulBatteryLevelUpdate: 7
The Heart Rate Notification is sent successfully, Heart Rate = 156.
The Heart Rate Notification is sent successfully, Heart Rate = 168.
SimulBatteryLevelUpdate: 8
The Heart Rate Notification is sent successfully, Heart Rate = 180.
SimulBatteryLevelUpdate: 9
The Heart Rate Notification is sent successfully, Heart Rate = 192.
The Heart Rate Notification is sent successfully, Heart Rate = 204.
SimulBatteryLevelUpdate: 10
The Heart Rate Notification is sent successfully, Heart Rate = 216.
SimulBatteryLevelUpdate: 11
The Heart Rate Notification is sent successfully, Heart Rate = 228.
The Heart Rate Notification is sent successfully, Heart Rate = 240.
SimulBatteryLevelUpdate: 12
The Heart Rate Notification is sent successfully, Heart Rate = 252.
```

4. Press '**d**' on any of the HyperTerminal to disconnect the devices.

The BLE Heart Rate Sensor can work with any other BLE-compatible device (for example, phone, tablet) with appropriate software (for example, Android, iOS with installed application which supports the Heart Rate Profile). For instance, you can use the CySmart mobile app (Android or iOS) as the Heart Rate Service Client:

Figure 15. CySmart iOS App

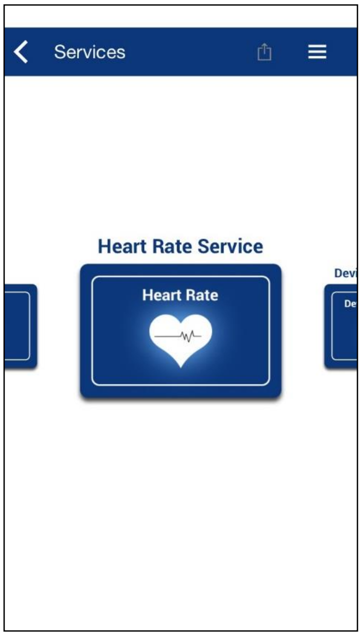
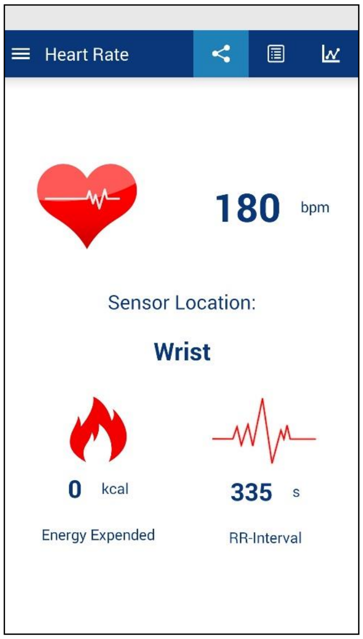


Figure 16. CySmart Android App



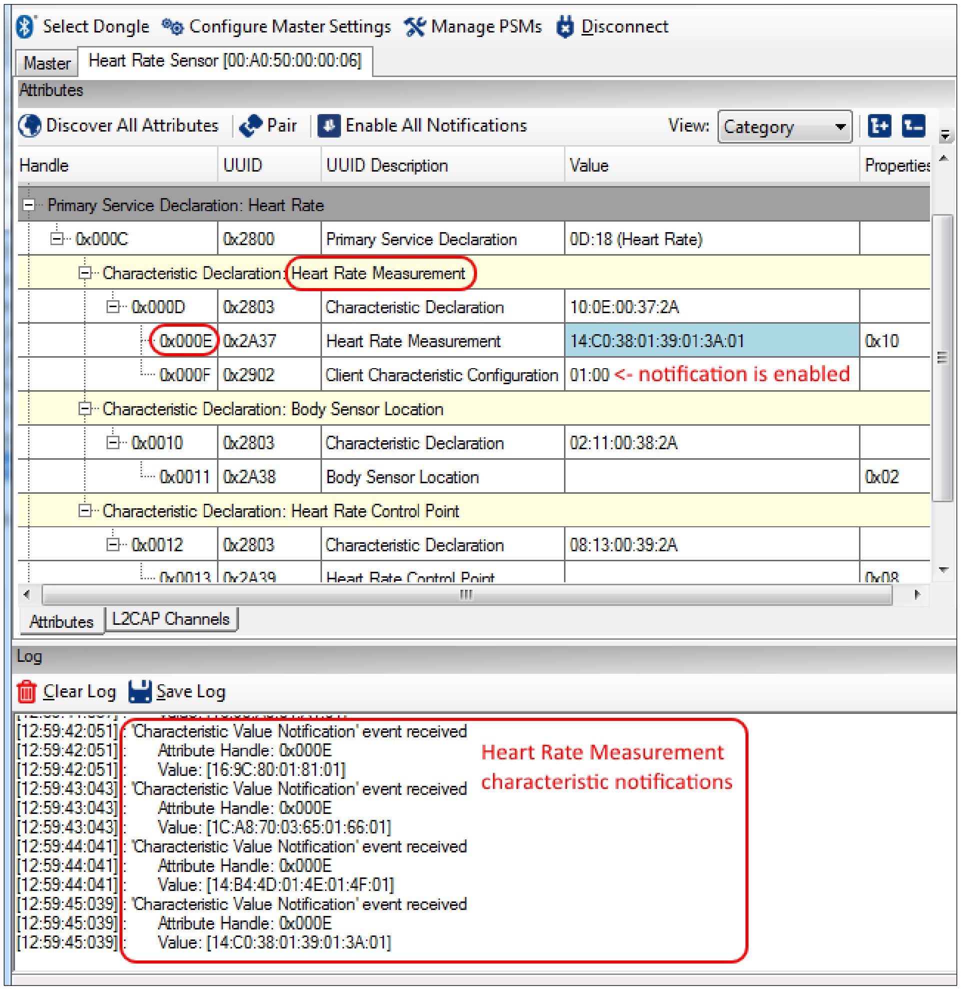
Also, the Heart Rate Sensor can be used with the CySmart app for Windows. You need to match the security settings between the Heart Rate Sensor and CySmart Client and pair (bond) the devices before writing (enabling notifications and so on) into the server's GATT database. For further instructions on how to use the CySmart application, see the CySmart User Guide.

A simple example on how to use the CySmart Windows application as the Heart Rate Service Client:

- Connect the CySmart BLE dongle to a USB port on the PC.
- Launch the CySmart app and select the connected dongle in the dialog window.
- Press **SW1** to reset the development kit and to start advertising.
- Click **Start Scan** to discover available devices.
- Select the Heart Rate Sensor from the list of available devices and connect to it.
- Click **Pair**, then **Discover All Attributes**, and **Enable All Notifications** in the CySmart app.

Observe the Heart Rate Measurement characteristic notifications with simulated data as shown in Figure 17.

Figure 17. Heart Rate Measurement Characteristic Notifications in CySmart



Handle	UUID	UUID Description	Value	Properties
Primary Service Declaration: Heart Rate				
0x000C	0x2800	Primary Service Declaration	0D:18 (Heart Rate)	
Characteristic Declaration: Heart Rate Measurement				
0x000D	0x2803	Characteristic Declaration	10:0E:00:37:2A	
0x000E	0x2A37	Heart Rate Measurement	14:C0:38:01:39:01:3A:01	0x10
0x000F	0x2902	Client Characteristic Configuration	01:00 <- notification is enabled	
Characteristic Declaration: Body Sensor Location				
0x0010	0x2803	Characteristic Declaration	02:11:00:38:2A	
0x0011	0x2A38	Body Sensor Location		0x02
Characteristic Declaration: Heart Rate Control Point				
0x0012	0x2803	Characteristic Declaration	08:13:00:39:2A	
0x0013	0x2A39	Heart Rate Control Point		0x08

```
[12:59:42:051]: 'Characteristic Value Notification' event received
[12:59:42:051]:     Attribute Handle: 0x000E
[12:59:42:051]:     Value: [16:9C:80:01:81:01]
[12:59:43:043]: 'Characteristic Value Notification' event received
[12:59:43:043]:     Attribute Handle: 0x000E
[12:59:43:043]:     Value: [1C:A8:70:03:65:01:66:01]
[12:59:44:041]: 'Characteristic Value Notification' event received
[12:59:44:041]:     Attribute Handle: 0x000E
[12:59:44:041]:     Value: [14:B4:4D:01:4E:01:4F:01]
[12:59:45:039]: 'Characteristic Value Notification' event received
[12:59:45:039]:     Attribute Handle: 0x000E
[12:59:45:039]:     Value: [14:C0:38:01:39:01:3A:01]
```

Heart Rate Measurement characteristic notifications

For more details on the Heart Rate Service characteristic data structures, see the HRS Specification.

Related Documents

Application Notes		
AN210781	Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes the PSoC 6 MCU with BLE Connectivity, and how to build a basic code example.
AN215656	PSoC 6 MCU Dual-Core CPU System Design	Presents the theory and design considerations related to this code example.
Software and Drivers		
CySmart – BLE Test and Debug Tool		CySmart is a BLE host emulation tool for Windows PCs. The tool provides an easy-to-use GUI to enable the user to test and debug their BLE 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 Programmable System-on-Chip		PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual (TRM)
Development Kit (DVK) Documentation		
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit		

Document History

Document Title: CE217639 - BLE Heart Rate with PSoC 6 MCU with BLE Connectivity

Document Number: 002-17639

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5968184	NPAL	11/20/2017	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

Cypress Developer Community

Forums | WICED IOT 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. 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.