CE225540 – CYW20819 Low Power

Objective

This example demonstrates low power modes on CYW20819 using ModusToolbox IDE.

Requirements

Tool: ModusToolbox™ IDE 1.1 or later version

Programming Language: C

Associated Parts: CYW20819

Related Hardware: CYW920819EVB-02 Evaluation Kit

Overview

This example demonstrates the use of low power mode APIs for the CYW20819 and shows how to measure current on the CYW20819EVB-02 Evaluation Kit. This example uses switch SW3 on the evaluation kit to switch between different states such as starting BLE advertisement, disconnecting from BLE connection, and so on. On startup, the device initializes the stack and immediately enters enhanced Power Down Sleep (ePDS) mode. Pressing switch SW3 will start advertisement while in ePDS mode. An external GAP central can be used to connect to this device and enable notifications. The device will maintain the connection while in ePDS mode. Upon disconnection, the device will enter HID-Off mode for 10 seconds and then start executing from the beginning.

Hardware Setup

Remove jumpers J14 and J18 to disable unused peripherals on the evaluation kit. For the rest of the jumpers, use the kit's default configuration. Refer to the kit guide to ensure the kit is configured correctly. The default settings will power the board at 3.3V. If you want to power the kit using a different voltage like 1.8V or using coin cell then you need to change the jumpers J7 and J8 to the appropriate position for VDDIO and VBAT domain.

An iOS/Android mobile device or a PC with CY5677 CySmart BLE 4.2 USB Dongle can act as the BLE Central which can connect to the Peripheral device CYW20819 on the CYW20819EVB-02 Evaluation Kit.

Software Setup

This code example consists of two parts: the GAP Central and the GAP Peripheral. For the GAP Central, download and install the CySmart app for iOS or Android. You can also use the CySmart Host Emulation Tool Windows PC application if you have access to the CY5677 CySmart BLE 4.2 USB Dongle.

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

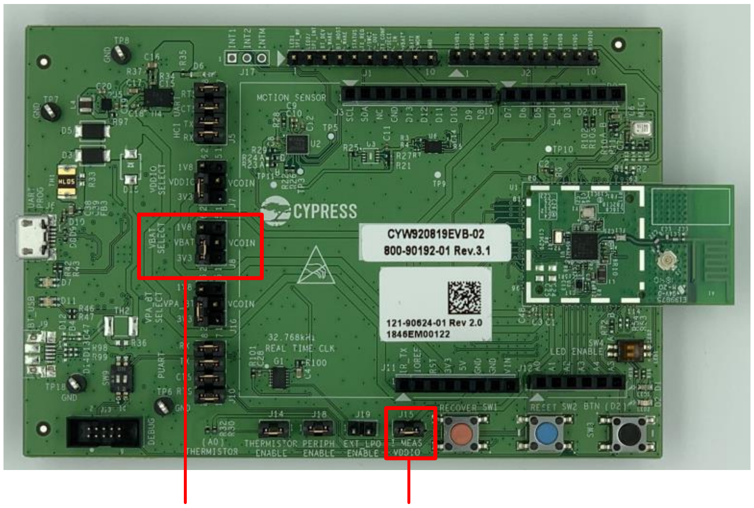
iOS Android

This example uses a terminal emulator. Install one if you don't have one. The instructions use Tera Term but you can use any terminal emulator that you prefer.

Operation

1. Connect the kit to your PC using the provided USB cable.
2. If you want to measure power consumption, connect an ammeter across J15.1 and J15.2, and a second ammeter across J8.2 and J8.4 to measure current in VDDIO and VBAT domains respectively as shown in Figure 1. If you don't have 2 ammeters, then measure current on one domain at a time. Note that the VPA_BT power domain is not used on this kit and hence there is no need to measure current on this domain.

Figure 1. CYW920819EVB-02 Jumpers to measure Current

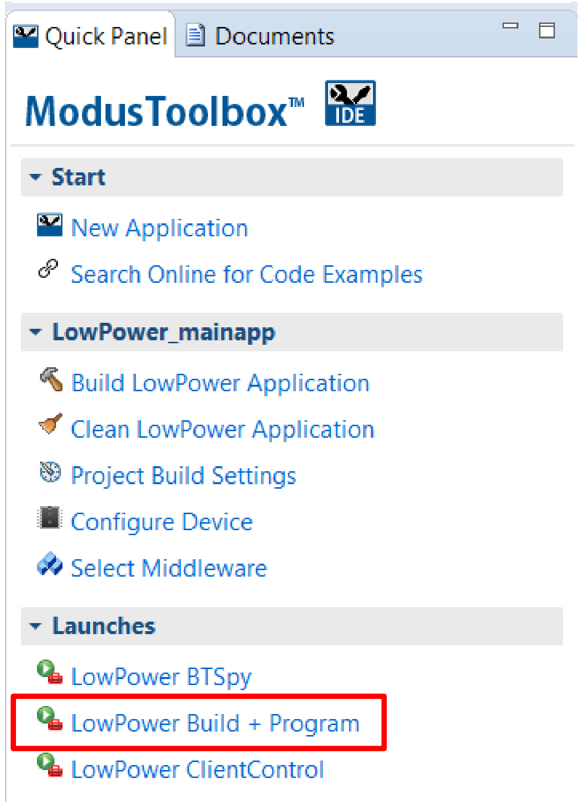


3. Remove jumpers J14 and J18 to disable unused peripherals on the evaluation kit.
4. The USB Serial interface on the kit provides access to the two UART interfaces of the CYW20819 device – WICED HCI UART, and WICED Peripheral UART (PUART). The HCI UART is used only for downloading the application code in this code example and the PUART is used for printing the Bluetooth stack and application trace messages. Open your terminal software and select the PUART COM port, with a baud rate setting of 115200 bps. If you want to disable the trace messages, then comment out the following line in the file low_power_20819.c:

```
wiced_set_debug_uart( WICED_ROUTE_DEBUG_TO_PUART );
```

5. Import the code example into a new or existing workspace. If you are not familiar with this process, see KBA225201.
6. **Build and Program the Application**: In the project explorer, select the **<App Name>_mainapp** project. In the Quick Panel, scroll to the **Launches** section, and click the **<App Name> Build + Program** configuration as shown in Figure 2.

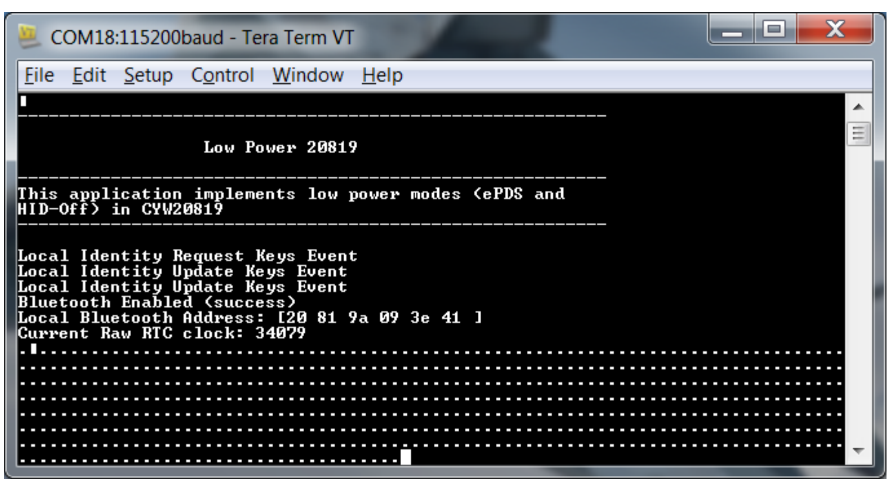
Figure 2. Programming the CYW20819 Device from ModusToolbox™



Note: If the download fails, it is possible that a previously loaded application is preventing programming. For example, the application may use a custom baud rate that the download process does not detect or the device may be in low power mode. In that case, it may be necessary to put the board in recovery mode, and then try the programming operation again from the IDE. To enter recovery mode, first, press and hold the Recover button (SW1), press and release the Reset button (SW2), and then release the Recover button (SW1).

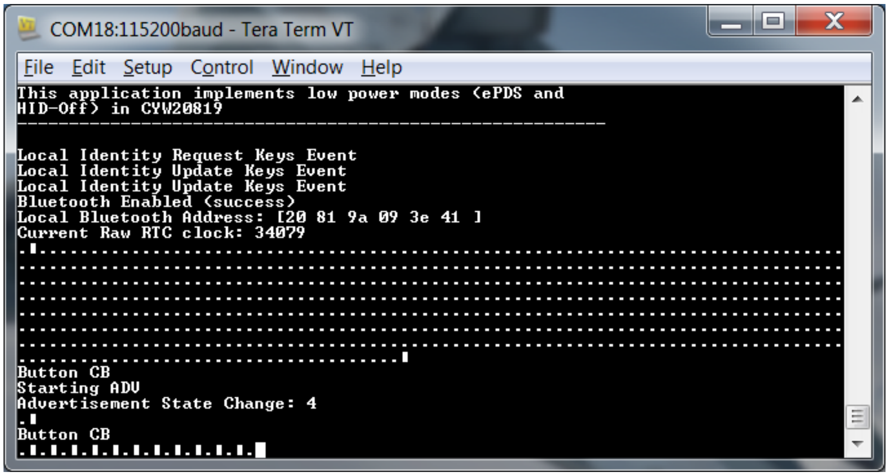
7. After the programming is complete the device will boot up and enter ePDS mode. The continuous '**.**' prints means that the PMU is asking for permission from the application to enter ePDS mode. The '**.**' stops when the device enters ePDS mode. Note the current readings from the ammeters. These are the current consumed in ePDS mode with no Bluetooth activity.

Figure 3. Bootup Log



8. Press switch SW3 on the evaluation kit. The application will get a button callback and it will start advertising. Note the current readings on the ammeters. This is the average current in ePDS mode with advertisement. The '.' Character is printed every time the application gets a sleep permission callback from the PMU and the boxes are printed every time the wakes up from ePDS. These prints will not occur if you comment them out in the `low_power_sleep_handler` function.

Figure 4. Start Advertisement Log



9. To test a connection using the CySmart mobile app, follow the steps below (see equivalent CySmart app screenshots in Figure 5):
   a. Turn ON Bluetooth on your Android or iOS device.
   b. Launch the CySmart app.
   c. Swipe down on the CySmart app home screen to start scanning for BLE Peripherals; your device appears in the CySmart app home screen. Select your device to establish a BLE connection. Once the device is connected you can read the current numbers from the ammeters. These are the current in ePDS mode with a connection at a connection interval of 100 ms.
   d. Select the **GATT DB** from the carousel view.
   e. Select **Battery Service** and then select **Characteristics.**
   f. Select **Notify**. The CYW920819EVB-02 will start sending GATT notifications to the mobile device. Note the current readings on the ammeters. These are the current in ePDS mode with a connection at a connection interval of 100 ms and notifications being sent every 5 seconds.
   g. Disconnect the Bluetooth connection by pressing SW3 on the kit or by backing out from the mobile app. The device will enter HID-Off mode for 10 seconds. Note the current numbers. These are the current numbers in HID-Off mode.

Figure 5. Evaluating with the CySmart App on Android

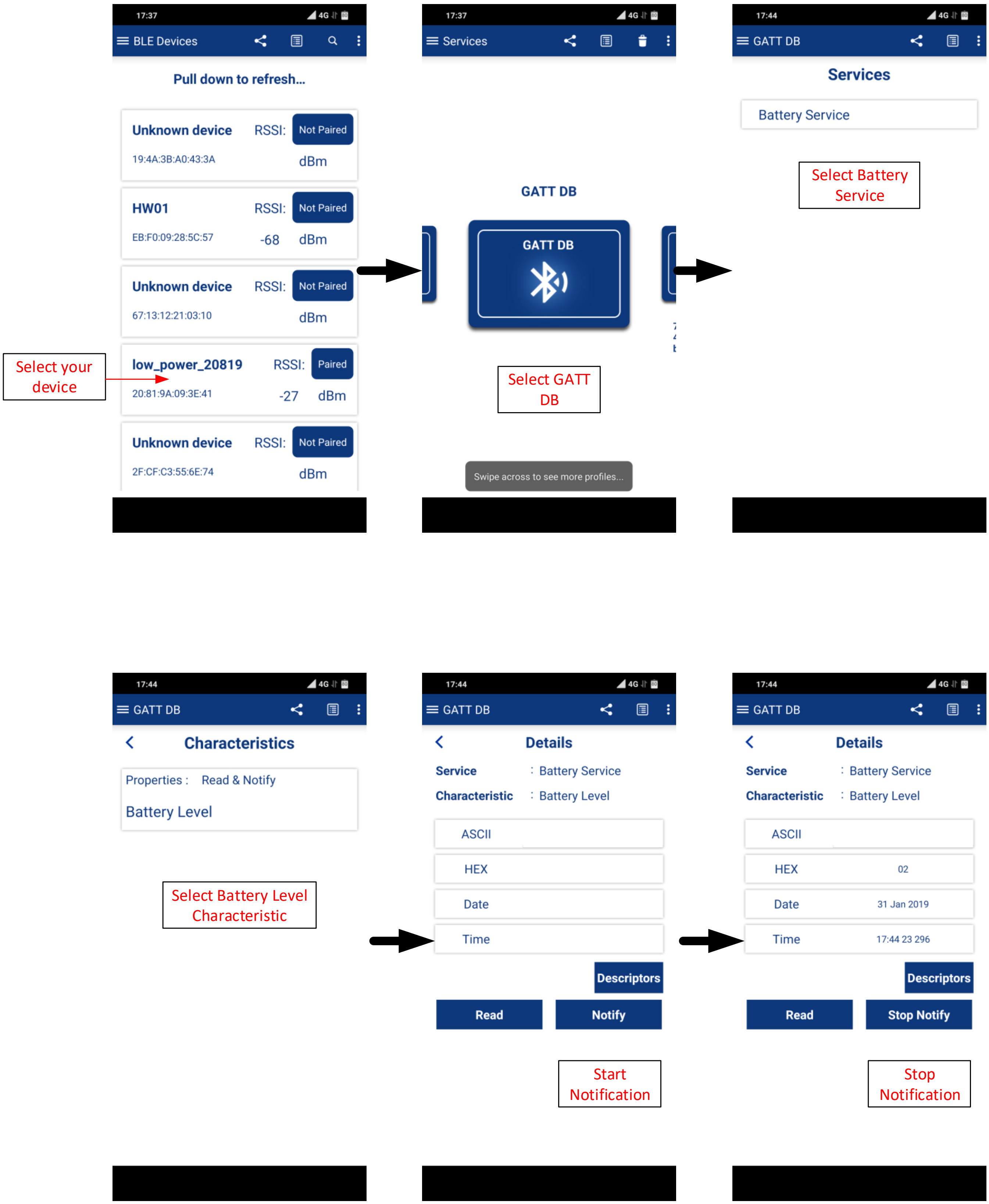


Figure 6. Connection, Pairing, and Connection Parameters Update Messages

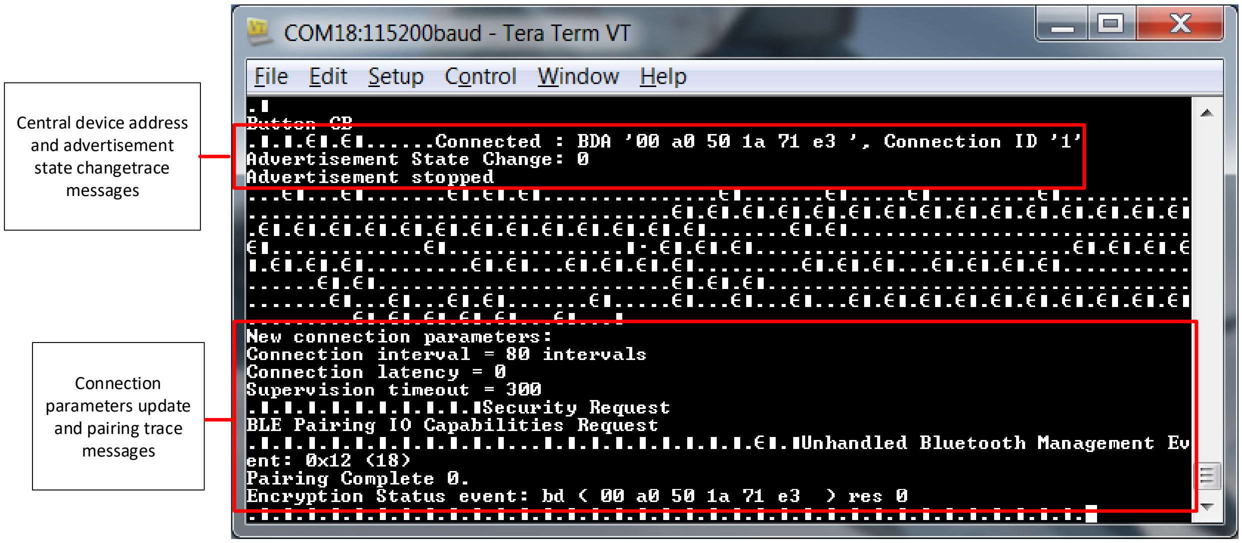
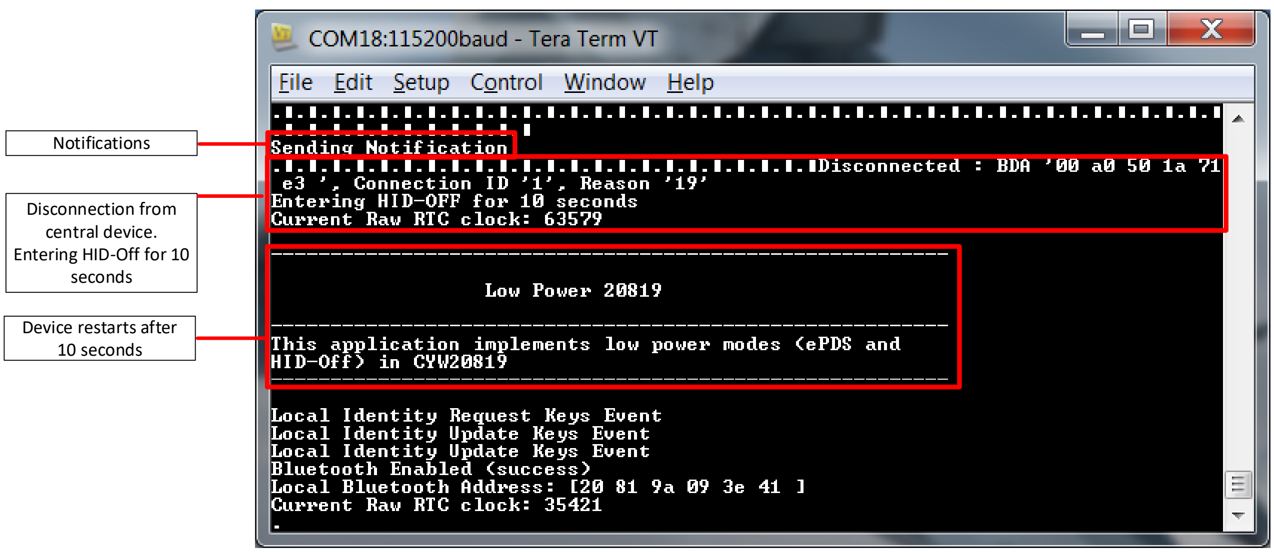


Figure 7. Disconnection, HID-Off, and Restart Trace Messages



10. To test using the CySmart desktop application on a PC:
    a. Open the CySmart desktop application and connect to the CySmart CY5677 dongle (Central device).
    b. Refer to the CySmart User Guide on how to use this application.
    c. **Scan** and **Connect** to 'low_power_20819' device. When asked for a connection parameter update, accept it. After the connection is established, you can measure the current values. These are the current numbers in ePDS mode with connection at 100 ms interval.
    d. Go to the device tab and click **Discover all attributes**.
    e. Click on **Enable all Notifications**. The device will start sending notifications every 5 seconds. Note the current readings on the ammeters. These are the current in ePDS mode with a connection at a connection interval of 100 ms and notifications being sent every 5 seconds.
    f. Click **Disconnect** to disconnect from the Central. The device will enter HID-Off mode for 10 seconds. Note the current numbers. These are the current numbers in HID-Off mode.

Design and Implementation

This code example implements a GATT Server and GAP Peripheral role on CYW920819EVB-02. Once the device is powered on it boots up, configures sleep, initializes the Bluetooth stack, registers a button interrupt and GATT database and then enters ePDS mode. The user needs to press switch SW3 on the kit to start low duty advertisement. The device is still in ePDS mode. The user can now connect to the device using a GAP Central. Upon connection, the device will request connection parameters to be updated (specifically the connection interval to 100 ms). If the request is accepted, then the connection interval changes to 100 ms. The device remains in ePDS mode and maintains the connection by sending empty packets. The GAP central can now discover all attributes and enable GATT notifications. The peripheral will start sending a dummy battery level value every 5 seconds.

The GATT server implements a Battery Service with a Battery Level characteristic. This characteristic is readable and notifiable.

The application code and the Bluetooth stack runs on the Arm® Cortex®-M4 core of the CYW20819 SoC. The application level source files for this code example are listed in Table 1.

Table 1. Code Example File Structure

File Name	Comments
low_power_20819.c	Contains the application_start() function which is the entry point for execution of the user application code after device startup. It also has the sleep callback function used by the PMU. Function to enter HID-Off is also in this file. The contents in this file can be referenced to implement low power modes in other applications.
wiced_bt_cfg.c	These files contain the runtime Bluetooth stack configuration parameters like device name, advertisement/connection settings etc.
cycfg_bt.h, cycfg_gatt_db.c, cycfg_gatt_db.h	These files reside in the “GeneratedSource” folder under the application folder. They contain the GATT database information generated using the Bluetooth Configurator tool.
low_power_20819_ble.c	This file contains the Bluetooth events callback function along with other functions to service Bluetooth events. It also contains the button callback function

Application Flow

The following diagrams shows the flow of the application code. Figure 8 shows the flow of the application when it boots up, Figure 9 shows the flow of the button callback, Figure 10 shows the flow of the bt stack management event callbacks, Figure 11 shows the flow of the GATT event callbacks and Figure 12 shows the tree of the functions that are called on the BT and GATT event callbacks from the stack.

Figure 8. Application Flow after Bootup

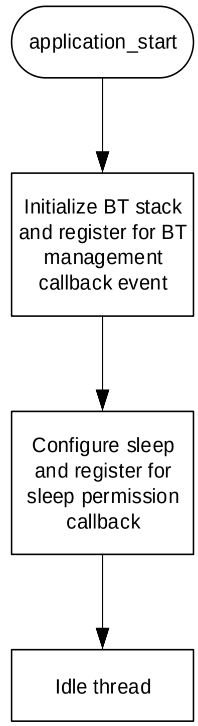


Figure 9. Button Callback Flow

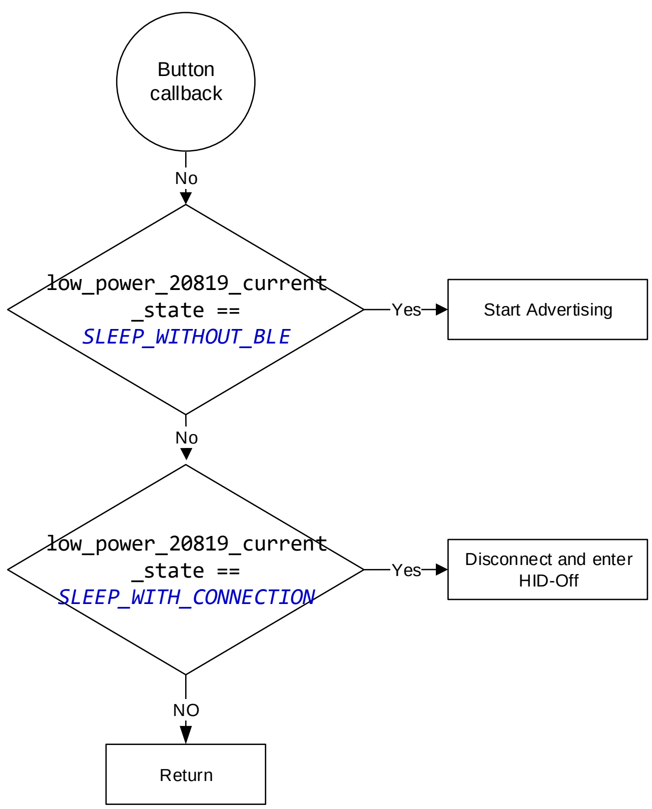


Figure 10. BT Stack Management Callback Flow

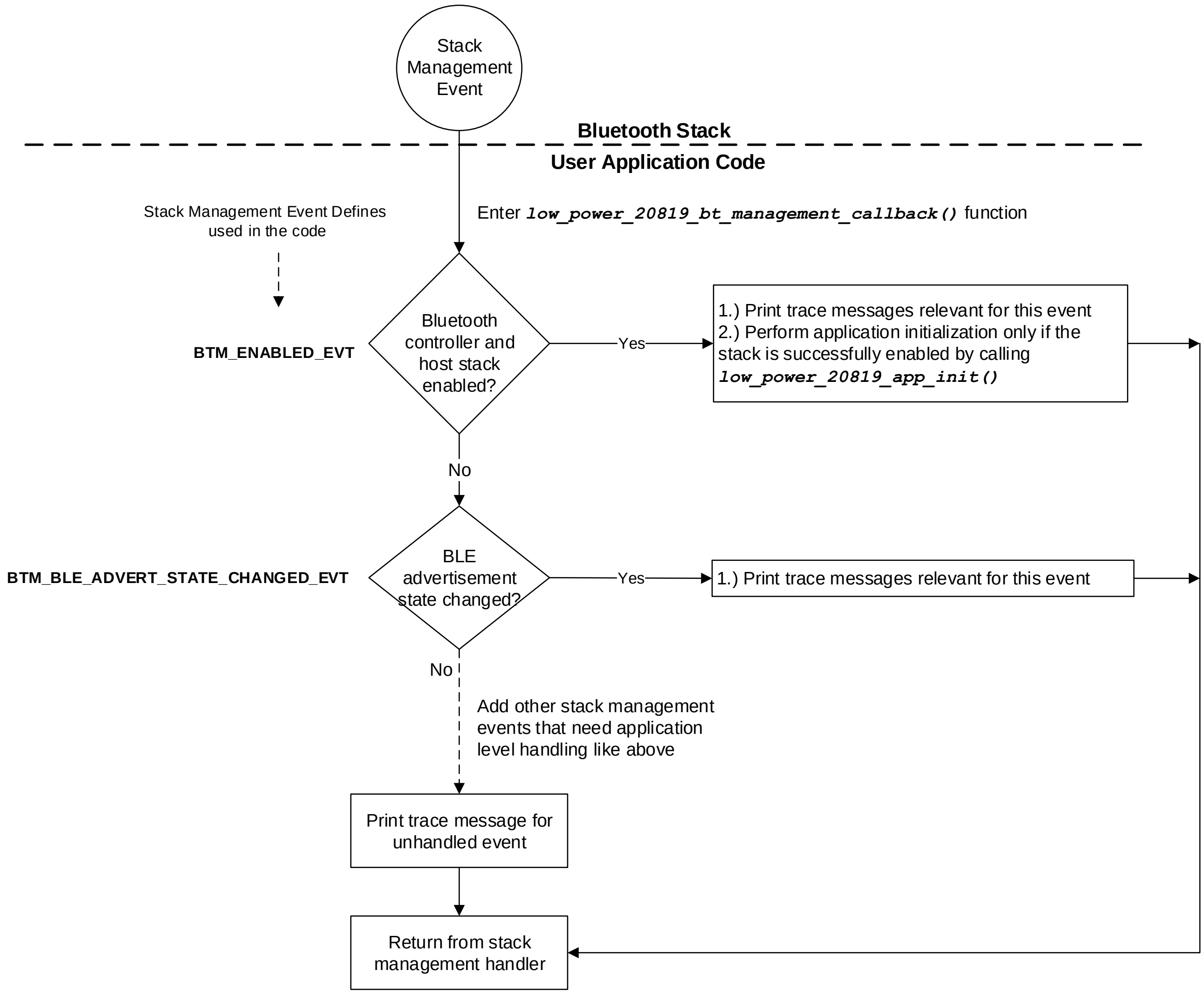


Figure 11. GATT Event Callback Flow

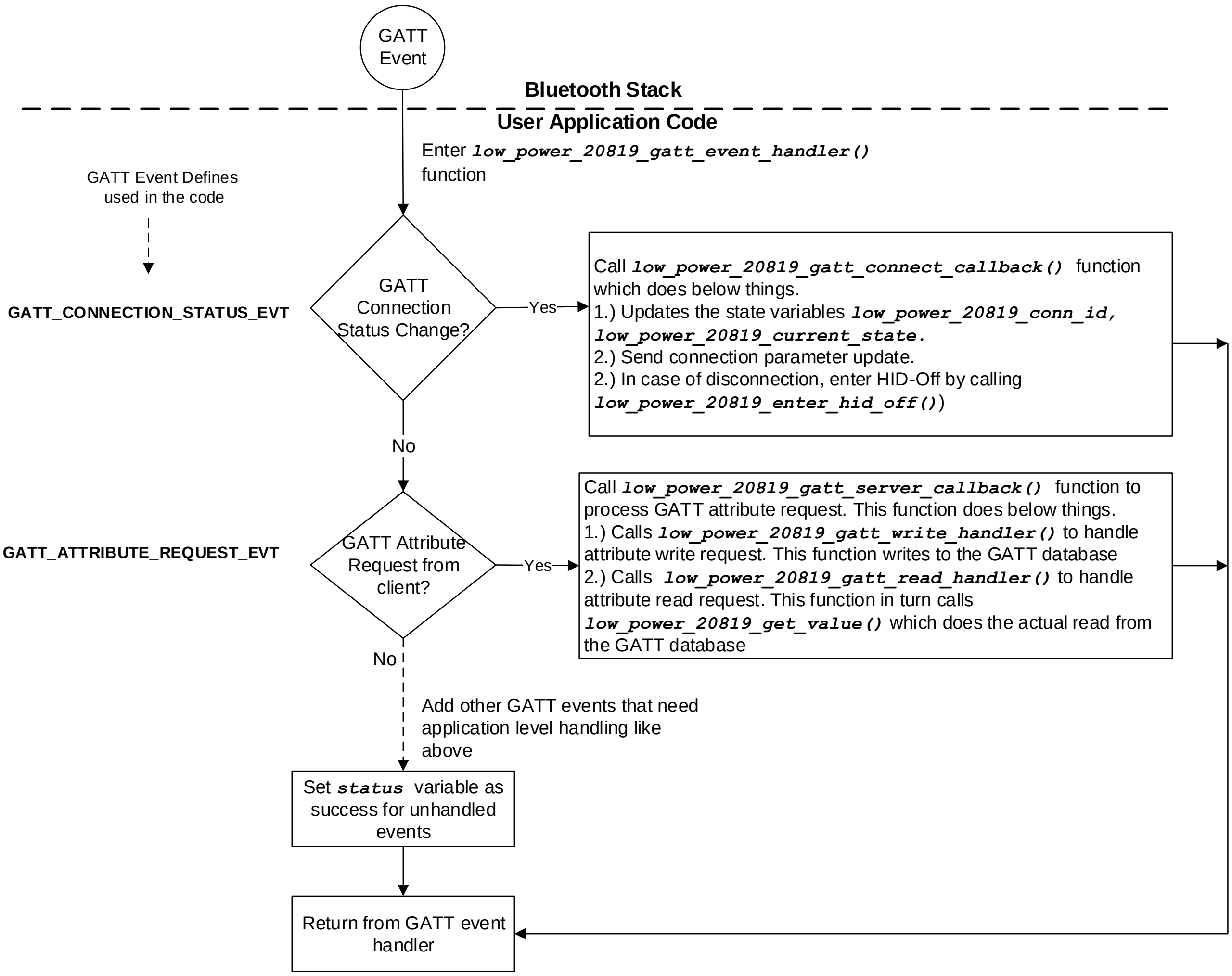
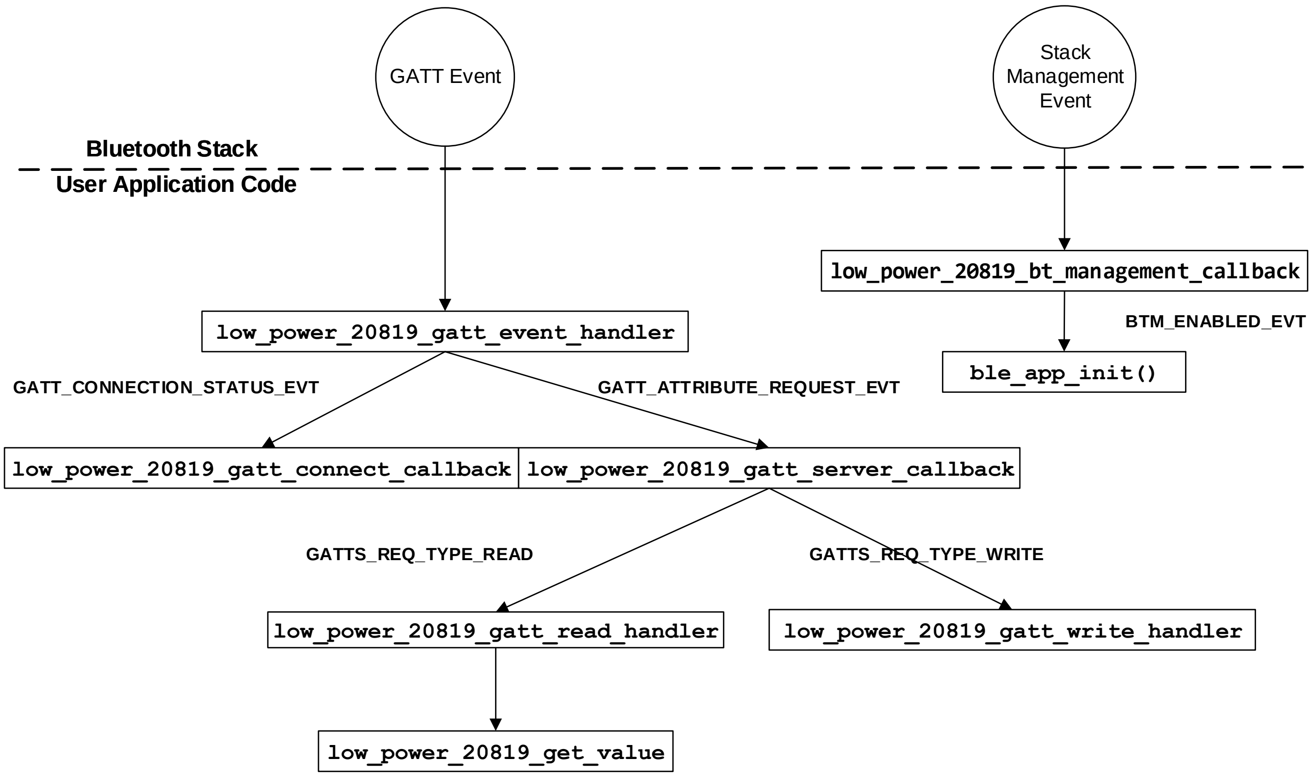


Figure 12. BT Management and GATT Events Function Call Tree



Current Measurements

The instantaneous current consumed by the device is not a steady state value but varies depending on the state of the chip that dynamically changes with the power-mode transitions. Therefore, it is practically impossible to measure each individual instantaneous current with a handheld multimeter because the duration of these current bursts is very small. Therefore, you should use a multimeter that provides the option to set the ”aperture” of the measurement. The aperture is the period ”T” during which the multimeter measures the instantaneous currents, integrates them, and then displays the average current for the period ”T”. For accurate measurements, the aperture of the multimeter should be set to be the same as the advertising or the connection interval. The following tables gives the current values for VBAT and VDDIO in various scenarios. Note that the current is averaged over 10 second intervals.

Table 2. CYW20819 Current in Different Modes

State	ePDS Enabled		ePDS Disabled	
	VDDIO	VBAT	VDDIO	VBAT
No Bluetooth activity	2.1 µA	7.7 µA	47.9 µA	0.97 mA
ADV (2.56 seconds interval)	2.3 µA	26.1 µA	47.9 µA	0.98 mA
Connection (100 ms connection interval)	3.2 µA	147.2 µA	47.9 µA	1.02 mA
Notifications (5 s interval)	3.3 µA	148.3 µA	47.9 µA	1.02 mA

Table 3. Current in HID-Off Mode

State	VDDIO	VBAT
HID-Off	2.2 µA	0.7 µA

Connection Interval	ePDS Enabled		ePDS Disabled	
	VDDIO	VBAT	VDDIO	VBAT
7.5 ms	14.5 µA	1.49 mA	47.9 µA	1.58 mA
10 ms	11.3 µA	1.16 mA	47.9 µA	1.43 mA
11.25 ms	10.2 µA	1.03 mA	47.9 µA	1.38 mA
12.5 ms	9.4 µA	0.92 mA	47.9 µA	1.34 mA
13.75 ms	9.9 µA	0.96 mA	47.9 µA	1.31 mA
15 ms	9.4 µA	0.89 mA	47.9 µA	1.28 mA
25 ms	6.6 µA	0.54 mA	47.9 µA	1.17 mA
50 ms	4.4 µA	0.27 mA	47.9 µA	1.08 mA
100 ms	3.2 µA	0.14 mA	47.9 µA	1.03 mA
500 ms	2.31 µA	0.04 Ma	47.9 µA	0.98 mA
1000 ms	2.2 µA	0.02 mA	47.9 µA	0.98 mA
2000 ms	2.2 µA	0.02 mA	47.9 µA	0.98 mA
4000 ms	2.1 µA	0.02 mA	47.9 µA	0.97 mA

Note that these current values also include some leakage current on the board as some GPIOs connected to the on-board components draw current. For accurate current numbers refer the device datasheet.

Resources and Settings

This example uses the default device configurator settings i.e. when this example is imported to ModusToolbox, the IDE creates the file design.modus (used for design configuration with default settings for the Kit. Note that in the design.modus file, the SPI and I2C modules are enabled but since these are not used in the application they will not cause any current leakage. It also provides the GATT database files so the user doesn’t have to generate the files.

Reusing This Example

This example is designed in a way so that the user can use the low power functions from this example in his own example with minimal changes.

Related Documents

Application Notes	
AN225684 - GETTING STARTED WITH CYW20819	Describes CYW20819 device and how to build your first ModusToolbox project
Code Examples	
Visit the Cypress GitHub repo for a comprehensive collection of code examples using ModusToolbox IDE	
Development Kit Documentation	
CYW20819EVB-02 Evaluation Kit	
Tools Documentation	
ModusToolbox IDE	The ModusToolbox cross-platform IDE simplifies development for IoT designers. Look in *<ModusToolbox install>/docs*.

Cypress Resources

Cypress provides a wealth of data at www.cypress.com to help you to select the right device, and quickly and effectively integrate the device into your design.

Document History

Document Title: CE225540 – CYW20819 Low Power

Document Number: 002-25540

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6486072	AMKA	02/20/2019	New code example

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Arm® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

Cypress Developer Community

Community | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

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