

TI Designs Bluetooth® Smart to Wi-Fi® IoT Gateway



TI Designs

TI Designs provide the foundation that you need including methodology, testing and design files to quickly evaluate and customize the system. TI Designs help you accelerate your time to market.

Design Resources

[TIDC-BLE-TO-WIFI-IOT-GATEWAY](#)

[CC2650](#)

[CC3200](#)

[TPS79601](#)

[TPD2EUSB30](#)

TI Design Files

Product Folder

Product Folder

Product Folder

Product Folder



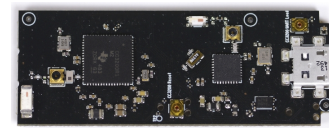
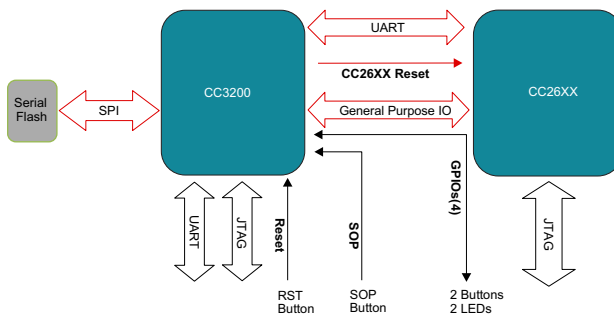
[ASK Our E2E Experts](#)
[WEBENCH® Calculator Tools](#)

Design Features

- Connect *Bluetooth*® Smart Devices To IoT Cloud
- USB Powered, Small Form Factor, Low-Power Wi-Fi® Connection
- USB (Over UART) Command Interface For Configuration—Works With SensorTag
- HTTP Server and Pages For Configuration
- Single-Board *Bluetooth* Smart Wi-Fi Integrated Design
- Message Queue Telemetry Transport (MQTT) Protocol Enabled For IoT Connectivity
- Over-The-Air (OTA) Firmware Update

Featured Applications

- Internet of Things (IoT)
- Social Alerts
- Health and Fitness
- Remote Tracking



An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

1 Key System Specifications

Table 1. Key System Specifications

PARAMETER	SPECIFICATION	DETAILS
Wi-Fi chip	A Single-Chip Wireless MCU with integrated Wi-Fi network processor and power management subsystem	Refer to the following link: http://www.ti.com/product/cc3200
Bluetooth Smart chip	The CC2650 is a wireless MCU targeting Bluetooth Smart, ZigBee™ and 6LoWPAN, and ZigBee RF4CE remote control applications	Refer to the following link: http://www.ti.com/product/cc2650
Power chip	The TPS79601 low dropout (LDO) low-power linear voltage regulator features ultralow-noise and excellent line and load transient responses in small outline, 3 × 3 VSON package	Refer to the following link: http://www.ti.com/product/tps79601/description
ESD chip	The TPD2EUSB30 are 2 channel Transient Voltage Suppressor (TVS) based Electrostatic Discharge (ESD) protection diode arrays	Refer to the following link: http://www.ti.com/product/TPD2EUSB30
Antenna	2450AT18D0100 for Bluetooth Smart and 2450AT42B100 for Wi-Fi. 2.4 Ghz, ceramic chip antenna	Refer to the following links: http://www.johansontechnology.com/datasheets/antennas/2450AT18D0100.pdf http://www.johansontechnology.com/datasheets/antennas/2450AT42B100.pdf
V _{IN}	Input voltage 5 V, USB powered	
V _{OUT}	Output voltage 3.3 V, powers the gateway	

1.1 Deliverables

The Bluetooth Smart to Wi-Fi Gateway TI design comprises the following mentioned collaterals.

1.1.1 Hardware Design Files

The hardware design files help the customer design a custom board. The package includes schematics, layout, and Gerber files.

1.1.2 Software Source

The software package contains all the gateway source files. The software is modular, which makes it easier for the developer to include a particular module.

1.1.3 Collaterals

This design guide explains the usage of the gateway.

1.2 Trademarks

SimpleLink, Code Composer Studio, Internet-on-a-Chip are trademarks of Texas Instruments.
 Cortex is a trademark of ARM Limited.
 ARM is a registered trademark of ARM Limited .
 Cortex is a registered trademark of ARM Limited.
 Bluetooth is a registered trademark of Bluetooth SIG.
 Dropbox is a trademark of Dropbox Inc.
 IBM is a registered trademark of IBM Corporation.
 HyperTerminal is a registered trademark of Microsoft Corporation.
 Silabs is a trademark of Silicon Laboratories.
 Wi-Fi, Wi-Fi certified are registered trademarks of Wi-Fi Alliance.
 ZigBee is a trademark of ZigBee Alliance.
 All other trademarks are the property of their respective owners.

2 System Description

Figure 1 shows the gateway block diagram.

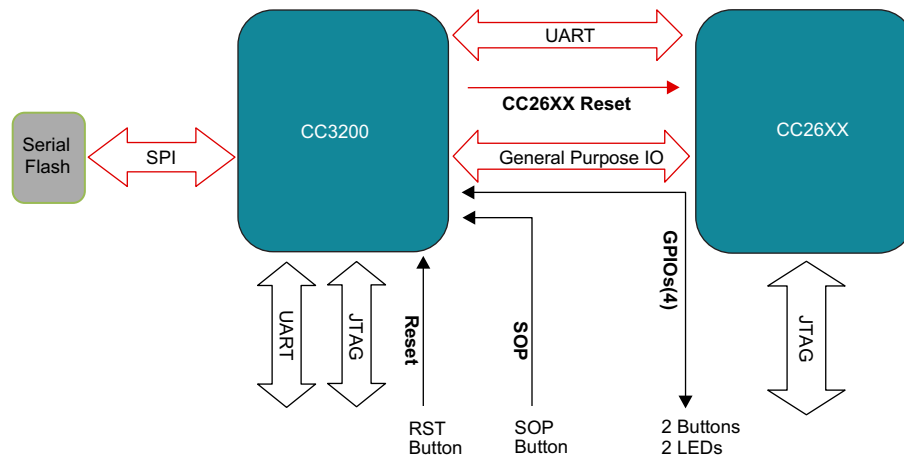


Figure 1. The Gateway Block Diagram

This *Bluetooth* Smart-to-Wi-Fi Gateway solution connects the *Bluetooth* Smart devices to the cloud over Wi-Fi. The integrated hardware solution comprises the CC3200 Wi-Fi wireless MCU and the CC2650 Wireless MCU on a single board. Customers may reuse the reference design to design their own gateway. The software and hardware design resources reduce engineering efforts, shorten time to market, and help developers and customers release their products with cloud connectivity features faster.

The CC3200 Wireless MCU is an intelligent, low-power Wi-Fi chip with an ARM® Cortex®-M4 application processor. The CC3200 contains a wireless network processor with stable and certified TCP / IP stack integrated on a single chip. The device comes with on-chip security protocols and various network application protocols (mDNS and others). CC3200 is the first and only chip in the world to be Wi-Fi certified® by the Wi-Fi alliance at chip level.

The CC3200 SDK (Software Development Kit) facilitates easy development experience for the user. The CC3200 contains CCS, IAR and GCC toolchain support, app notes and extensive user guides. The SimpleLink™ APIs provide an easy and well abstracted access to the stack functionality. The SDK provides MQTT support (for IoT), OTA (Over The Air) download support, and numerous MCU examples, which reduces the development effort and thereby reduces the time to market. Extensive E2E support from TI helps the developers at every stage.

CC2650 is an ultralow-power wireless MCU capable of running multiple protocols (*Bluetooth* Smart, ZigBee, and 6LowPAN). This wireless MCU can run for years even when powered by coin-cell batteries or any energy-harvested sources.

The CC2650 platform comes ready-to-use, royalty free, and with certified wireless protocol stacks, TI RTOS, Code Composer Studio™ integrated development environment (IDE), development tools, online training, and E2E community support.

The CC3200 Wireless MCU powers the entire board and also communicates and controls the CC2650 *Bluetooth* Smart Wireless MCU over UART. With the support of its built-in Wi-Fi network processor, the CC3200 Wireless MCU ensures seamless connection with the remote cloud. The CC2650 Wireless MCU can connect to multiple *Bluetooth* Smart devices.

Two LED indicators are provided to indicate the status. Two buttons are provided to trigger various actions. A simple button is provided to switch the CC3200 to UARTLoad mode. Debugging and programming options for both CC3200 and CC2650 chips are provided by dedicated JTAG ports.

The gateway can easily be configured for the first time using the Command Line Interface (CLI). UniFlash flashes the binaries on CC3200.

A 1-MB Serial Flash (SFLASH) is provided for storing the binaries, and configuration files.

A dedicated RESET button is provided to bring the gateway out of unknown conditions during development.

2.1 CC2650—Wireless MCU

Figure 2 shows the CC2650 block diagram.

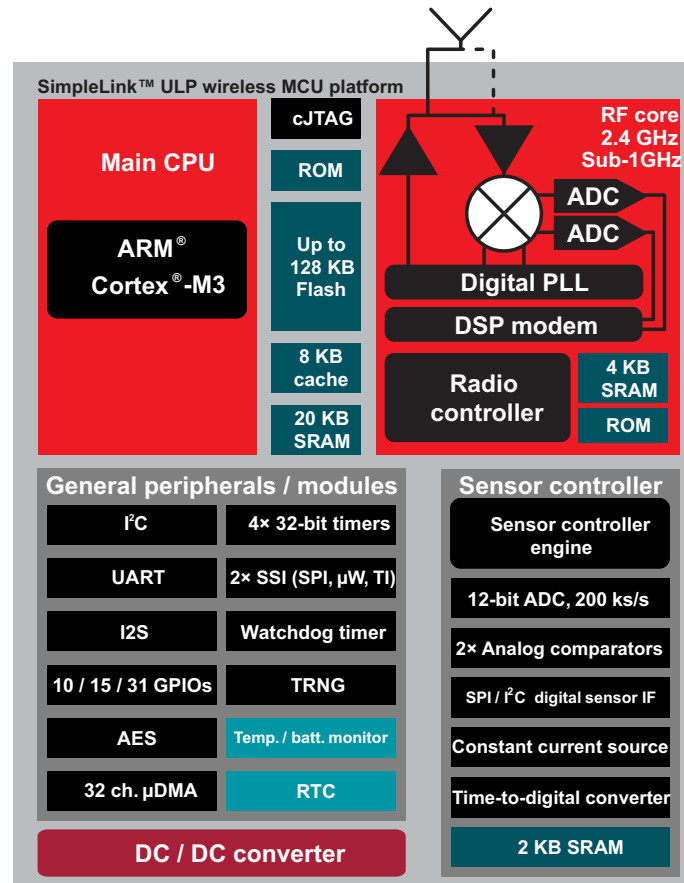


Figure 2. CC2650 Block Diagram

The CC2650 is a wireless-MCU-targeting *Bluetooth* Smart application. The device is a member of the CC26xx family of cost-effective, ultralow-power, 2.4-GHz RF devices. Very low active RF current consumption and very low MCU current consumption provide excellent battery lifetime and allows operation on small-coin cell batteries and in energy-harvesting applications. The CC2650 contains a 32-bit ARM Cortex™ M3 running at 48-MHz as the main processor and a rich peripheral feature set. This set includes a unique ultralow-power sensor controller, which is ideal for interfacing external sensors and collecting analog and digital data autonomously while the rest of the system is in sleep mode.

CC2650 is a network processor in the gateway design. For more details of CC2650, see [the product page](#).

2.2 CC3200—Wireless MCU

The CC3200 MCU subsystem contains an industry-standard ARM Cortex M4 core running at 80 MHz. The device includes a wide variety of peripherals, including a fast parallel camera interface, I²S, SD and MMC, UART, SPI, I²C, and four-channel ADC. The CC3200 family includes flexible embedded RAM for code and data, and ROM with an external serial flash bootloader and peripheral drivers.

Figure 3 shows the CC3200 block diagram.

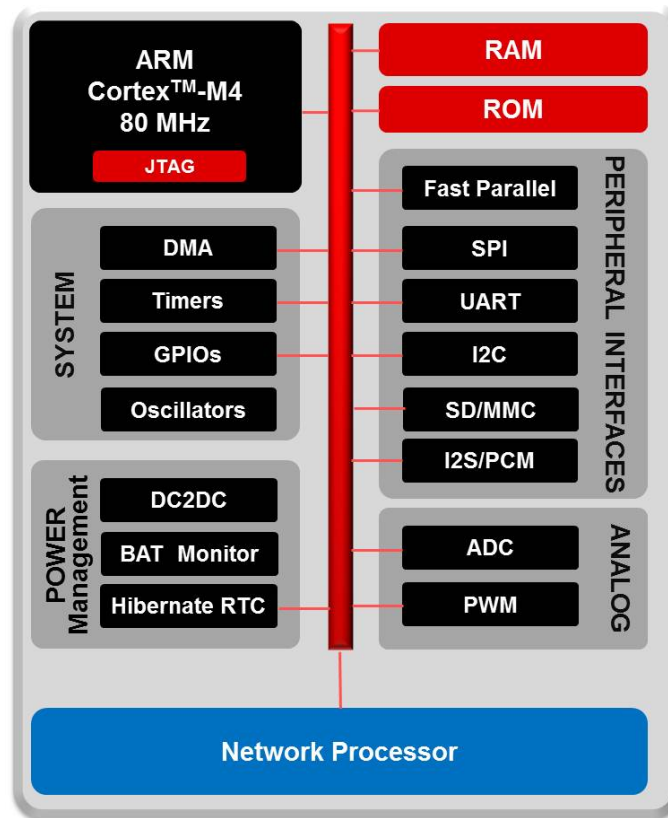


Figure 3. CC3200 Block Diagram

The Wi-Fi network processor subsystem features a Wi-Fi Internet-on-a-Chip™ and contains an additional dedicated ARM MCU that completely offloads the applications MCU. This subsystem includes an 802.11 b/g/n radio, baseband, and MAC with a powerful crypto engine for fast, secure Internet connections with 256-bit encryption. The CC3200 device supports Station, Access Point, and Wi-Fi Direct modes. The device also supports WPA2 personal and enterprise security and WPS 2.0. The Wi-Fi Internet-on-a-chip includes embedded TCP/IP and TLS/SSL stacks, and multiple internet protocols.

The CC3200 is the host processor in the gateway design. For more details about CC3200, see the [product page](#).

2.3 TPS79601—Power IC

The TPS796 family of low-dropout (LDO), low-power linear voltage regulators features high power-supply rejection ratio (PSRR), ultralow-noise, fast startup, and excellent line and load transient responses in small-outline, 3 × 3 VSON, SOT223-6, and TO-263 packages. Each device in the family is stable with a small, 1-μF ceramic capacitor on the output. The family uses an advanced, proprietary BiCMOS fabrication process to yield extremely low dropout voltages (for example, 250 mV at 1 A). Each device achieves fast start-up times.

For more details of TPS79601, see the [product page](#).

2.4 TPD2EUSB30 ESD Protection Diode

The TPD2EUSB30 is a 2-channel Transient-Voltage-Suppressor (TVS)-based Electrostatic-Discharge (ESD)-protection diode array. The TPDxUSB30/A devices are rated to dissipate ESD strikes at the maximum level specified in the IEC 61000-4-2 international standard (Contact). These devices also offer 5-A (8 / 20 μs) peak pulse current ratings per IEC 61000-4-5 (Surge) specification.

For more details of TPD2EUSB30, see the [product page](#).

3 System Design Theory

The *Bluetooth Smart-to-Wi-Fi* IoT gateway aims at addressing the problem statement *How can the data on a Bluetooth Smart device be available over the internet or on a Wi-Fi enabled device.* Figure 4 shows the gateway system.

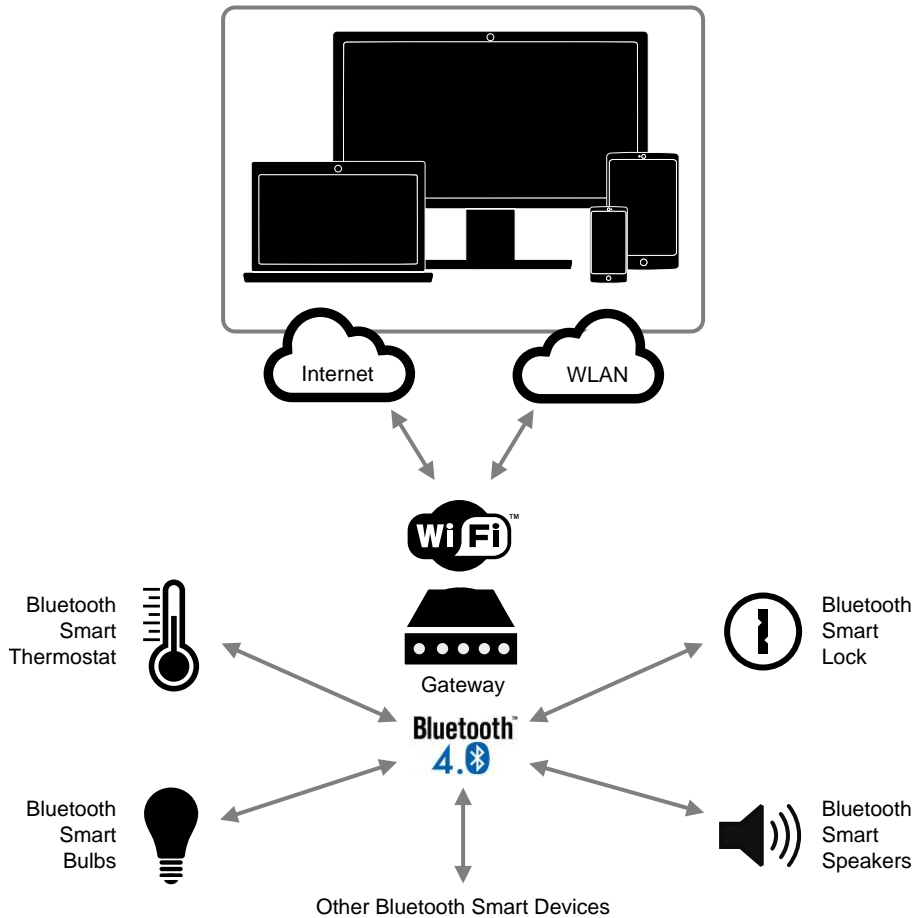


Figure 4. Gateway System

SensorTag is one of TI’s most popular designs. The *Bluetooth Smart*-enabled version is usually limited to using a phone or tablet as a gateway, which limits the use in more infrastructure-oriented applications as the phone may not always be present. Given the emergence of the Internet of Things (IoT), these devices should be connected to the Internet or a private LAN seamlessly (that is, without changing the software or design of the existing SensorTag modules).

3.1 Architecture

Figure 5 shows the software architecture.

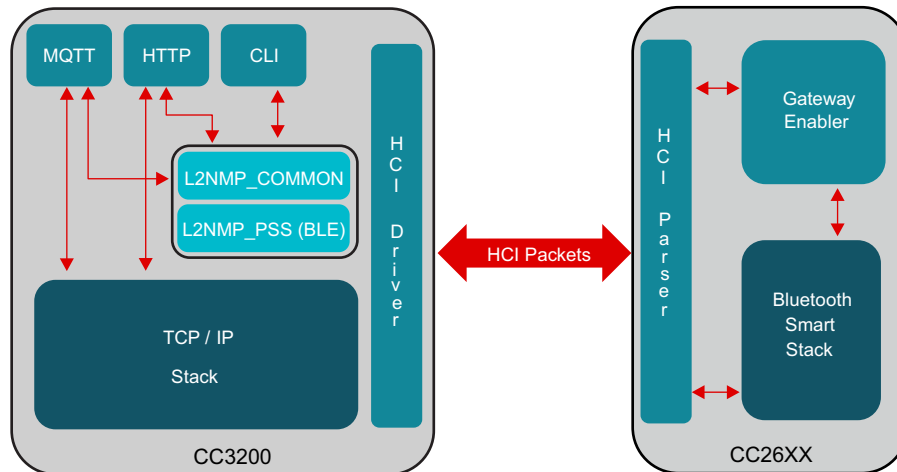


Figure 5. Software Architecture

The CC3200 Wireless MCU acts as a master and controls the CC26XX *Bluetooth* Smart Wireless MCU. The Gateway enabler application on the CC26XX brings out the *Bluetooth* Smart stack functionalities over the host controller interface (HCI) commands.

The application runs on the CC3200 and contains the following modules:

- L2NMP_COMMON (generic interface with upper layer agent **and** CLI, L2NMP authentication and database)
- L2NMP_PSS (BLE) (central and BLE database)
- TCP / IP stack (network processor)
-
- IoT stack (MQTT client stack—IBM®)
- Host HCI layer
- Utilities such as CLI and JSON parser
- OTA and serial boot loader (SBL) module

3.2 Working Principle

The software is a multitask architecture. The initialization sequence shown in Figure 6 spawns the tasks and executes various module-initialization functions.

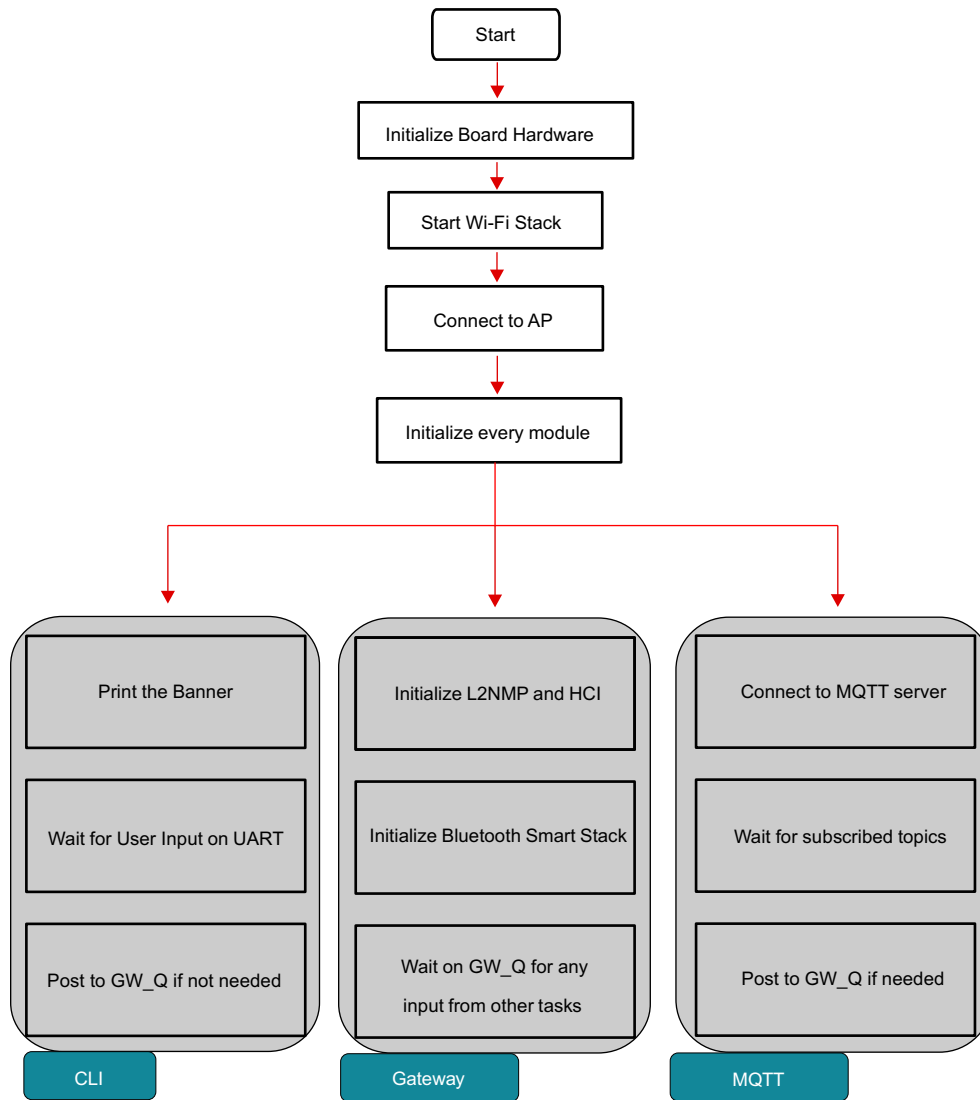


Figure 6. Initialization Sequence on CC3200 MCU

When configured, the gateway acts as a data pipe between the *Bluetooth* Smart devices and the monitoring entity (on Wi-Fi). No special configuration or code change is needed on the *Bluetooth* Smart device to remote it to the gateway. [Figure 7](#) shows the data flow.

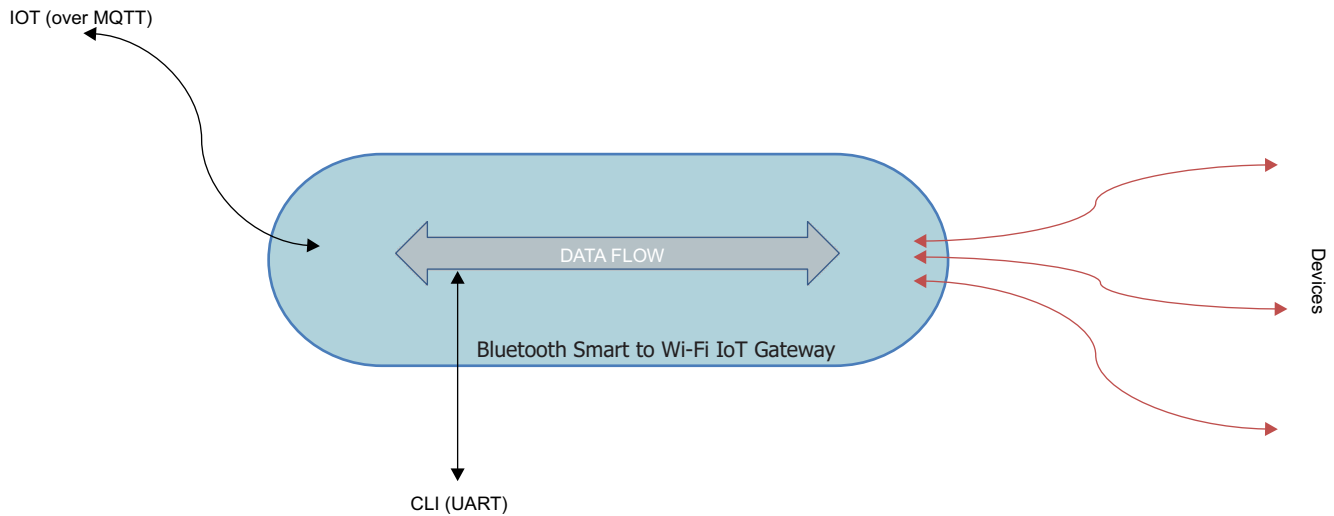


Figure 7. Data Flow

The gateway setup comprises of WLAN connection and *Bluetooth* Smart device connection. When the setup is complete, you can monitor the characteristics of the device over MQTT or CLI. With the software architecture, users can easily remove a particular entity from the binary (to reduce the size).

OTA download functionality is also present in the gateway. OTA can update the binary and also the Gateway Enabler binary.

4 Getting Started Hardware

The primary feature of the gateway is to make the characteristics of a connected *Bluetooth* Smart device available on the cloud (IoT using MQTT).

Starting the gateway hardware is an easy process. To power on the gateway, connect it to a PC or a wall-mount USB adapter through the USB cable provided in the package. Connecting the gateway to a PC provides access to the command line interface on a terminal (TeraTerm or HyperTerminal®), which aids in controlling or monitoring the gateway. When connected, press the RESET button to start the gateway.

Figure 8 shows the details of the board.

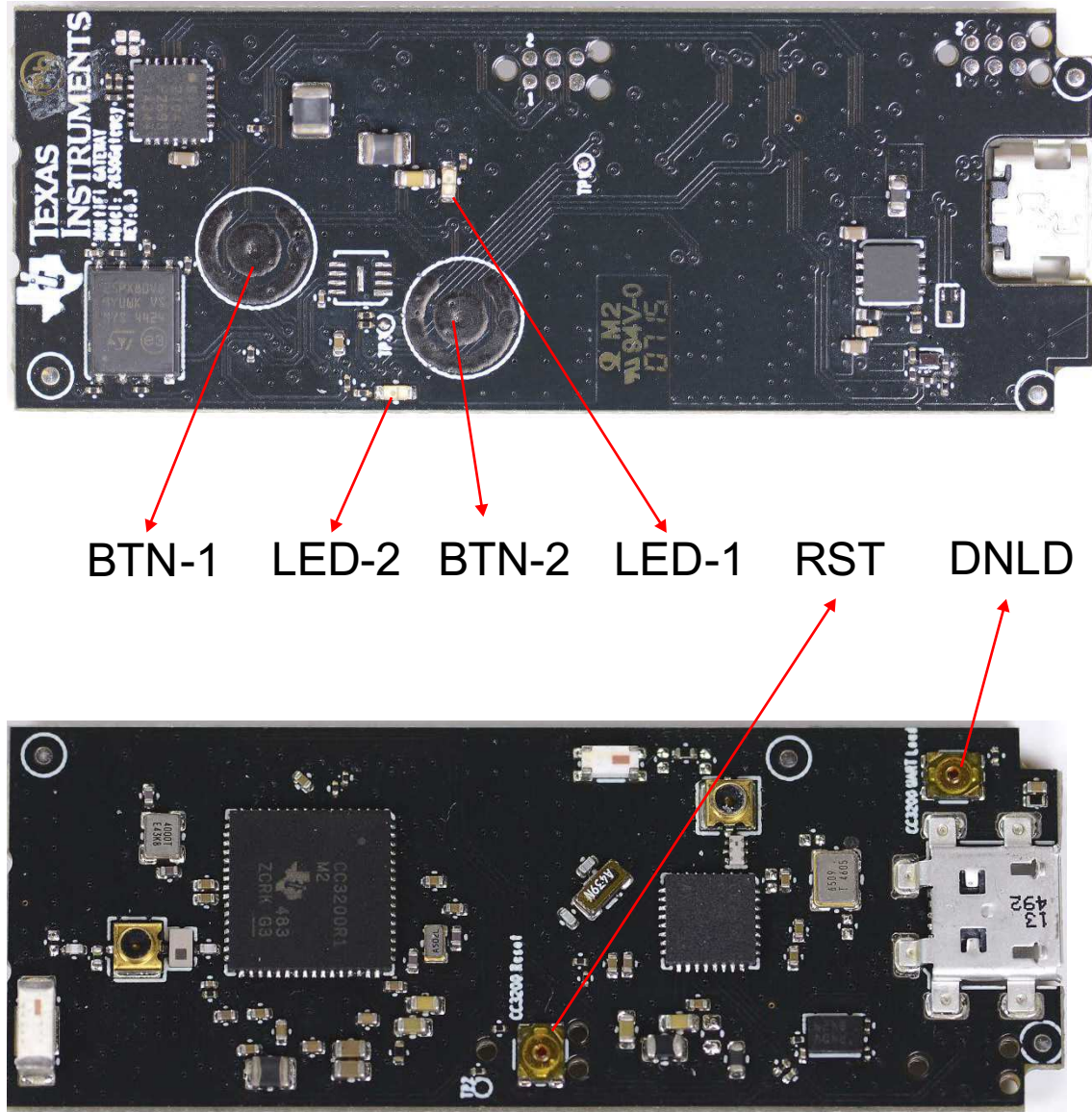


Figure 8. Board Details

4.1 Push Buttons and LEDs

Table 2. Button Description

REFERENCE	USAGE	COMMENTS
RST	Reset	This button is used to reset the gateway
DNLD	UartLoad	This button, when pressed along with RST, puts the CC3200 in UARTLOAD mode. This mode updates the firmware
BTN-1	Scan	This button is used to trigger the scan for <i>Bluetooth</i> Smart devices
BTN-2	Over The Air (OTA) trigger	This button triggers the OTA download of binaries

Table 3. LED Description

REFERENCE	INDICATION	COMMENTS
LED-1 (<i>Bluetooth</i> Smart status indicator)	OFF	<i>Bluetooth</i> Smart stack is UP
	ON	<i>Bluetooth</i> Smart stack Initialization in progress
	SLOW BLINK (1 per sec)	Scanning in progress
	FAST BLINK (2 per sec)	System Fatal ERROR
LED-2 (Wi-Fi status indicator)	OFF	AP connection success
	ON	AP connection failure
	SLOW BLINK (1 per sec)	SmartConfig in progress
	FAST BLINK (2 per sec)	System Fatal ERROR

4.2 Prerequisites

1. [Bluetooth Smart to Wi-Fi IoT gateway hardware](#)
2. Micro-USB cable
3. Access point with internet uplink available. The gateway must be connected to this access point.
4. Laptop or PC (Connected to the Internet for MQTT [Internet])
5. SensorTag (at least one) with default pre-programmed software
 - [CC26XX SensorTag](#)

4.3 Setup

Figure 9 shows a diagram of the setup.

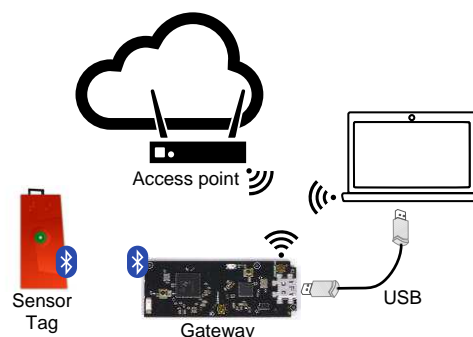


Figure 9. Setup Diagram

4.3.1 Laptop and PC Installations

- Download and install UniFlash (to flash CC3200 binaries and service pack) from the link here: <http://www.ti.com/tool/uniflash>
- Download and install TeraTerm: http://download.cnet.com/Tera-Term/3000-20432_4-75766675.html
- Download and install the latest Silabs™ virtual COM port driver for the CP2104 USB bridge IC (based on the OS) from the link here: <http://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx>.
 - The gateway COM ports appear in the device manager of the PC as Silicon Labs Standard COM CP210x USB to UART Bridge: Enhanced COM Port
- Download and install the latest CC3200 SDK and service packs from the link here: <http://www.ti.com/tool/cc3200sdk>

4.3.2 Access Point (AP) Configuration

Configure the access point in Open or WPA mode. If the AP is configured in WPA mode, write down the security key.

4.3.3 Steps

A typical usage of the gateway contains following steps:

1. Install the mentioned software on the laptop or PC
2. Connect the gateway to the laptop or PC using the USB cable
3. Configure the access point either in Open or WPA mode
4. Reset/Power on the gateway by pressing the Reset button
5. Connect the gateway to a Wi-Fi access point
6. Connect the gateway to SensorTag (using CLI or MQTT)
7. View, read, and write the capabilities of the SensorTag (using CLI or MQTT)

5 Getting Started Firmware

Download and install the binary package from [TIDC-BLE-TO-WIFI-IOT-GATEWAY](#).

5.1 Flashing the Binaries

The binary package contains these files:

- blefi.bin—The *Bluetooth* Smart to Wi-Fi IoT Gateway application that runs on CC320
- CC26xx.bin (gateway-enabler)—This application runs on CC2650.
- CC2650 SensorTag.sch [Schema files]
- Blefi.usf file (UniFlash Configuration file)

Follow these steps when using gateway for the first time.

1. Format the BleFi SFLASH
2. Download the Network Processor Service Pack
3. Download the binaries (open BleFi.usf and click program)

The TI UniFlash tool is used to format and download the gateway SFLASH. Refer to the [UniFlash](#) for more details.

5.1.1 SFLASH Formatting

Follow these steps:

1. Connect the gateway to laptop or PC equipped with UniFlash.
2. Change the COM PORT to the desired port number.
3. Press Format on the UniFlash tool.
4. Keep the DNLD button pressed and toggle the RST button on the gateway.
5. The previous procedure triggers the bootloader mode in CC3200, and the format should start.
6. The Successful Format message should appear on UniFlash.

5.1.2 Service Pack Download

The servicepack_1.0.0.1.2 service pack is provided through CC31xx_CC32xx_ServicePack-1.0.0.1.2-windows-installer.exe downloadable from <http://www.ti.com/tool/cc3200sdk> or <http://www.ti.com/tool/cc3100sdk>.

1. Download and install the service pack version servicepack_1.0.0.1.2.bin on a laptop or PC.
2. Connect the gateway to a laptop or a PC where UniFlash is installed.
3. Change the COM PORT to the desired port number.
4. Press Service Pack Programming on UniFlash tool.
5. Point to the path of the service pack binary in your computer.
6. Gateway—Keep DNLD button pressed and toggle the RST button.
7. The above procedure triggers the bootloader mode in CC3200, and the service pack programming should start.
8. The Successful message should appear on UniFlash.

5.1.3 Binary Download

1. Connect the gateway to a laptop or PC where UniFlash is installed.
2. Go to the folder where the binary package is present.
3. Double click on the blefi.usf file—This opens UniFlash.
4. Change the COM PORT to Desired Port Number
5. Press Program on UniFlash Tool.
6. Gateway—Keep DNLD button pressed and toggle the RST button.
7. The previous procedure triggers the bootloader mode in CC3200 and the download should start.
8. The Successful Download message should appear on UniFlash.

5.2 Building Source Code

This section describes the steps involved in building the blefi.bin, which executes on the CC3200 and the cc26xx.bin, which then executes on the CC2650.

5.2.1 Source Build (for CC3200 MCU)

1. Install Code Composer Studio IDE from the link [here](#).
2. Install the gateway installer.
3. By default, the package installs to the c:\ti folder.
4. Open Code Composer Studio and the following projects from the BleFi package.
 - l2nmp
 - l2nmp_ble
 - cli
 - ugateway
 - mqtt_app
 - npi
 - oslib
 - schema
 - ti_rtos_config
 - json
 - mqtt
 - ota
 - simplelink
5. Build clean all projects in sequence.
6. Upon successful build, find the ugateway.bin in the blefi/ccs_ugw/Release folder.

5.2.2 CC26xx Source Build (gw_enabler) Using CCS

1. Download BLE SDKv2.1 from <http://www.ti.com/ble-stack>.
2. Install CCS as in section 2.5.3 of *CC2640 Bluetooth low energy Software Developer's Guide (SWRU393)*.
3. Change the Radio settings in bleUserConfig.h in
 C:\ti\simplelink\ble_cc26xx_2_01_00_44423\Projects\ble\ICall\Include\bleUserConfig.h.

```
#elif defined( CC2650EM_5XD ) || defined( CC2650EM_4XD )
#define RF_FE_MODE_AND_BIAS ( RF_FE_DIFFERENTIAL |
RF_FE_INT_BIAS)
#elif defined( CC2650EM_4XS )
```
4. Change RAM boundary address in Linker Configuration file in
 C:\ti\simplelink\ble_cc26xx_2_01_00_44423\Projects\ble\HostTest\CC26xx\CCS\Config\CCSLinkerDefi
 nes.cmd --define=ICALL_RAM0_ADDR=0.
5. Open CCS and import the HostTestApp project (both HostTest and HostTestStack) as described
 in *CC2640 Bluetooth low energy Software Developer's Guide (SWRU393)*.
6. Check the Copy Project files to workspace option.
7. In CCS, open the Project Properties for the Application.
8. Change the board type in the last line of the Include Options under the CCS Build at
 workspace_v6_1\HostTest\FlashROM
 "\${TI_RTOS_DRIVERS_BASE}/ti/boards/SRF06EB/CC2650EM_5XD.
9. Select Projects → Build All options to build both the projects.
10. Find the HostTest.hex file under %CCS_WORKSPACE%\workspace_v6_1\HostTest\FlashROM.
11. Find the HostTestStack.hex under
 %CCS_WORKSPACE%\workspace_v6_1\HostTestStack\FlashROM.
12. See [Section 5.2.4](#) to merge the hex files and to convert them into cc26xx.bin.

5.2.3 CC26xx Source Build (gw-enabler) Using IAR

1. Install the IAR IDE from the [link](#).
2. Install the latest CC26xx SDK from the [link](#).
3. Open the *HostTest* project from the IAR IDE.
4. Choose CC2650 application.
5. Right-click on the project name → options → C/C++ Compiler.
6. Select *Preprocessor* tab.
7. Make the following changes.
8. In the *Additional include directories* section, make the following changes:
 1. Remove \$TI_RTOS_DRIVERS_BASE\%ti\boards\SRF06EB\CC2650EM_7ID
 2. Add \$TI_RTOS_DRIVERS_BASE\%ti\boards\SRF06EB\CC2650EM_5XD
9. In *Defined Symbols* section, make the following changes.
10. Add the preprocessor CC2650EM_5XD .
11. Make changes to the *bleUserConfig.h*, found in the location of the installed SDK. The file can be found at C:\ti\simplelink\ble_cc26xx_2_XX_XX\Projects\ble\ICall\Include
12. In ble_cc26xx_2_XX_XX\Projects\ble\ICall\Include\bleUserConfig.h Change

```
#define RF_FE_MODE_AND_BIAS (RF_FE_DIFFERENTIAL | RF_FE_EXT_BIAS)
```

 To

```
#define RF_FE_MODE_AND_BIAS (RF_FE_DIFFERENTIAL | RF_FE_INT_BIAS)
```
13. Compile the *Host Test App* project (both application and stack). (Two hex files are generated for application and stack.)
14. Refer [Section 5.2.4](#) to merge the hex files and convert them into cc26xx.bin.

5.2.4 Merging the Hex Files to CC26xx.bin

To create the binary file in the following procedure, a USB cable, SmartRF06 board, and a CC2650EM board are required.

1. Connect the SmartRF06+CC2650EM board to a PC where Flash Programmer 2 is installed using the USB cable.
2. Open Flash Programmer 2.
3. Connect to the SmartRF06+CC2650EM board.
4. Perform Mass erase.
5. Program Stack hex image.
6. Program App hex image.
7. Navigate to Edit tab of the Flash Programmer.
8. Read the entire flash contents.
9. Navigate to 0x1ffd8
10. Change the contents of 0x1ffd8, 0x1ffd9, 0x1ffda, and 0x1ffdb to c5, 03, fe, and c5, respectively (for the Serial Bootloader).
11. Click the *Write*.
12. Click *Save to File*.
13. Save as cc6xx.bin.

5.3 WLAN Connection

The gateway must be connected to a Wi-Fi access point (AP), which has cloud connectivity, to realize the IoT functionality.

CC3200 maintains Wi-Fi profiles, which helps in automatic connections to an AP during the boot up. The user should not need to make an AP connection in every power cycle. Figure 10 shows the WLAN connection process during initialization.

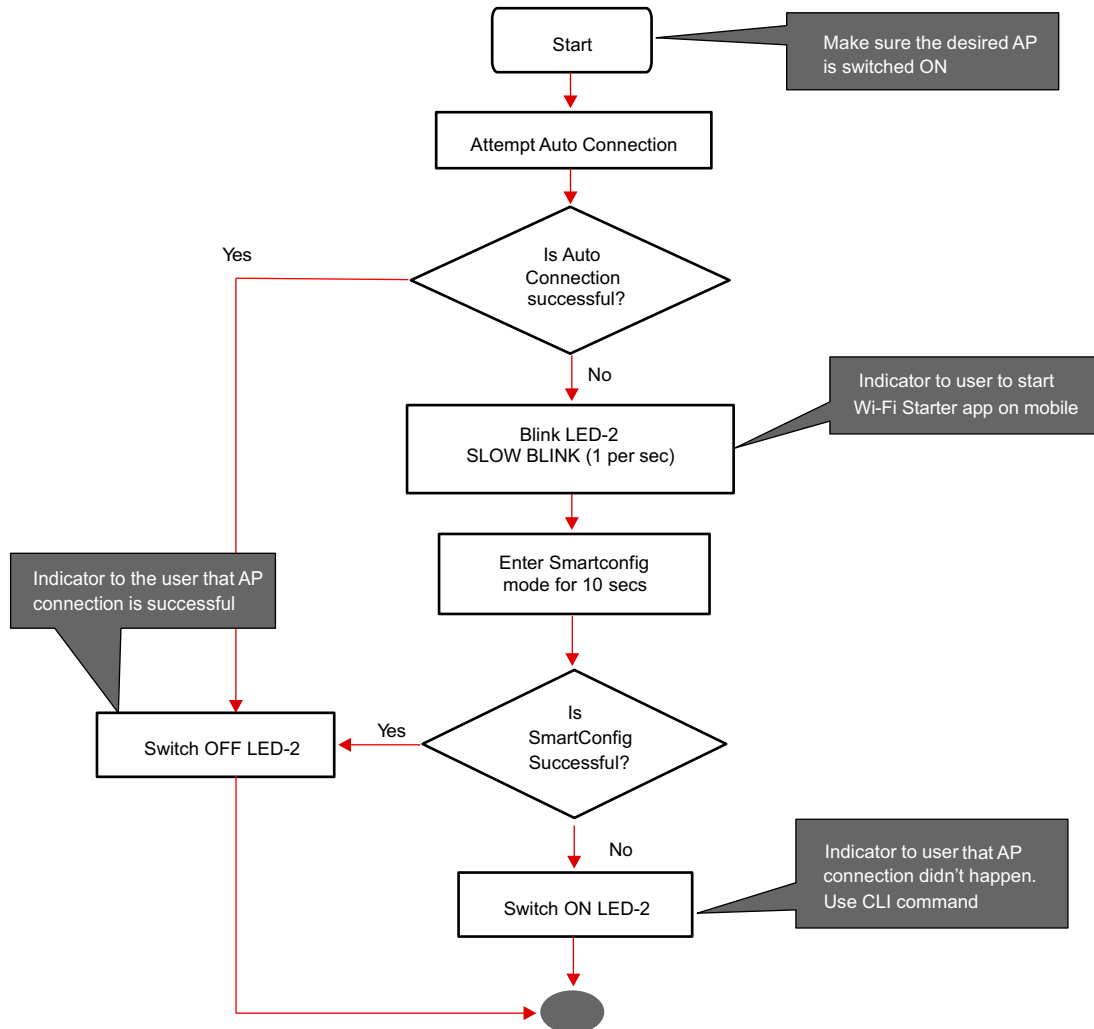


Figure 10. WLAN Connection Process During Initialization

The AP connection may not be successful during the bootup due to one of these reasons:

- No WLAN profile present
- AP not switched on before the gateway is reset
- The Smartconfig application was not available or was not successful

Post initialization, use the CLI command (`wlan_connect`; refer to [Section 6.1](#)) to connect to the AP. Ensure the AP is on while using this command. Upon successful connection, this configuration is stored as a WLAN profile. During subsequent power-cycles, this profile is checked if an auto connection is possible.

5.4 Bluetooth Smart Device Connection

A *Bluetooth* Smart device can be connected to the gateway using various UI commands (Refer to [Section 6](#)). When connected, the device is paired with the gateway, and the subsequent power cycle of the gateway does not need a UI command to be entered for connection.

5.4.1 Schema File

Typically, the characteristic in a *Bluetooth* Smart device is addressed using a handle. The handle is a numeric value and is not user friendly. The schema file mechanism in gateway maps the numeric handle to a human-readable string. This string can be used while monitoring the respective characteristic or while changing the value of the characteristic.

For example, to access the temperature data of the connected *Bluetooth* Smart Device,

```
G:>get 0 /Temp/Data
```

is used instead of

```
G:>get 0 15
```

Where 15 is the handle value.

NOTE: If schema file is not present, then default strings will be mapped (Example: char1).

To avail this facility, the user must create and download a schema file to the gateway. The schema file format is a human-readable JSON format and can be generated using any JSON editor. The SensorTag schema files in the package are created using <https://www.jsoneditoronline.org/>.

The name of the schema file is same as the device name that it publishes while advertising. An extension `.sch` must be added before downloading.

Example: SensorTag.sch

5.4.1.1 Schema File Format

```

{
  "Service_1 UUID": [
    {
      "Service_1.Characteristic_1 UUID": " Service_1.Characteristic_1 Name"
    },
    {
      "Service_1.Characteristic_2 UUID": " Service_1.Characteristic_2 Name"
    },
    ....
    ....
    ....
  ],
  "Service_2 UUID": [
    {
      "Service_2.Characteristic_1 UUID": " Service_2.Characteristic_1 Name"
    },
    {
      "Service_2.Characteristic_2 UUID": " Service_2.Characteristic_2 Name"
    },
    ....
    ....
    ....
  ],
  ....
  ....
  ....
}

```

6 Gateway Usage

This section assumes that the binaries are flashed in the gateway and the WLAN connection is also successful. For more information on the setup, refer to [Section 4.3](#).

Typical gateway usage contains

- WLAN connection
- Connection to a *Bluetooth* Smart device
- Data monitoring over MQTT/CLI

The gateway can be controlled or monitored using the UIs (User Interfaces)

- CLI over UART
- MQTT from a remote client

Table 4 summarizes the various control and monitoring capabilities of the UIs.

Table 4. Gateway UI Capabilities

ACTION	CLI	MQTT
WLAN Connect	√	X
WLAN Disconnect	√	X
Bluetooth Smart Device Scan	√	X (autoscan)
List Bluetooth Smart Devices	√	√
Bluetooth Device Connect	√	X (autoconnect)
Bluetooth Device Disconnect	√	√
List Bluetooth Device Characteristics	√	X
Read Bluetooth Smart Device Characteristics	√	√
Write to Bluetooth Smart Device Characteristics	√	√
Reset MCU	√	X
OTA Configuration	√	X
Reset MQTT Client	√	X
Configure Max Devices	√	X
Configure Gateway in Proxy or Catche (HL Confirmation)	√	X
Configure Attachment Mode (ALL, Enabled, Authorized)	√	X
Manage the List of Authorized Devices (Manage db)	√	X
Retrieve Wi-Fi Mac of Gateway	√	X

6.1 CLI

The gateway can be controlled or monitored using the CLI over UART. To start CLI, open any terminal application on the laptop or PC that is connected to the gateway. Figure 11 shows the serial port setup for CLI.

Terminal Setup:

1. In your terminal application, open the serial port Silicon Labs CP210x USB to UART bridge [COM100].
2. Setup the serial port with baud rate 115200 as shown in Figure 11.

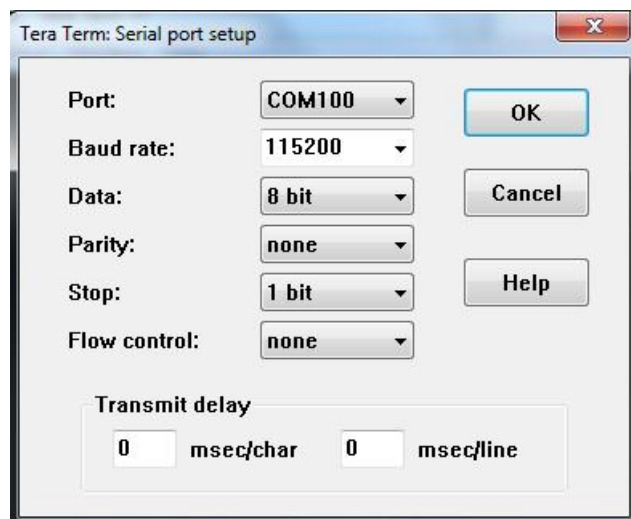


Figure 11. Serial Port Setup for CLI

The following sections describe the commands available in CLI.

6.1.1 HELP Command

help

help

Parameters	None
Description	help is used to display the CLI commands supported and their descriptions.
Examples	<ul style="list-style-type: none"> • help

6.1.2 LIST Command

list 0

list 0

Parameters	None
Description	list 0 lists attached and authorized devices.
Example	list 0

list 1

list 1

Parameter	None
Description	list 1 lists connected devices pending confirmation and authorization.
Examples	list 1

6.1.3 LINKT Command

linkt

linkt [device connection id]

Parameters	[device connection id] is the connection id of the connected <i>Bluetooth</i> Smart device from which to be disconnected. Use the list command to find the connection id of the device.
Description	linkt is used to disconnect from one of the connected <i>Bluetooth</i> Smart devices
Examples	linkt 0

6.1.4 LISTCHAR Command

listchar

listchar [device connection id]

Parameters	[device connection id] is the connection id of the connected <i>Bluetooth</i> Smart device to list the characteristics. Use the command to find the connection list id of the device.
Description	listchar is used to list the characteristics of one of the connected <i>Bluetooth</i> Smart devices.
Examples	listchar 0

6.1.5 GET Command

get

get [device connection id] [Charstring]

Parameters	[device connection id] is the connection id of the connected <i>Bluetooth</i> Smart device. Use the list command to find the connection id of the device. [Charstring] is the name of the characteristic to get.
Description	get is used to read the value of the characteristic of one of the connected <i>Bluetooth</i> Smart devices.
Examples	get 0 /Acc/Data get 0 char18

6.1.6 SET Command

set

set [device connection id] [Charstring] [length of value in bytes] [value]

Parameters	[device connection id] is the connection id of the connected <i>Bluetooth</i> Smart device. Use the list command to find the connection id of the device. [Charstring] is the name of the characteristic to be set. [length of value in bytes] is the length of the value to be written in bytes. [value] is the value of the characteristic to be written.
Description	set is used to write to the value of the characteristic of one of the connected <i>Bluetooth</i> Smart devices.
Examples	set 0 /Acc/Cfg 1 1 set 0 char18 1 1

6.1.7 WLAN_CONNECT Command

wlan_connect *wlan_connect [ssid] {key}*

Description wlan_connect is used to connect to an AP. When successfully connected, a WLAN profile is added. The gateway can be connected to the following kinds of AP.

- Open
- WPA

The Open AP connection does not require the key parameter. If the key parameter is present, a WPA connection is tried.

Parameters ssid is the string value. The ssid of the AP of the connection must be tried.
key is the security password for AP connection (only for WPA).

Examples wlan_connect open_ap_1
wlan_connect wpa_ap password123

6.1.8 WLAN_DISCONNECT Command

wlan_disconnect *wlan_disconnect*

Description wlan_disconnect is used to disconnect the WLAN connection to AP.

Parameters None

Examples

- wlan_disconnect

6.1.9 RESET Command

reset *reset*

Parameters None

Description reset is used to reset the gateway.

Examples

- reset

6.1.10 OTA Configuration Command

addotameta

addotameta [metadatastring]

Parameters	metadatastring – meta data string of the associated Dropbox™ account where the gateway OTA files are stored.
Description	addotameta is used to configure the Dropbox meta data string information used for OTA in gateway.
Examples	addotameta HNFGT_m-65YJJADDGGGBs7y8FWDvgBAbnVFzVUdhDxhVJHu7ung9dSFsc_dHO45

6.1.11 Bluetooth Smart Update Command

bleupdate

bleupdate

Parameters	None
Description	This command is used to update the CC26xx firmware. The update takes effect after a reboot of gateway hardware.
Examples	bleupdate

6.1.12 MQTT Gateway Mode Command

mqttagwmode

mqttagwmode mode <optional fields>

Parameters	[mode] – 1 for quickstart mode or 2 for registration mode
Description	This command is used to reset the MQTT mode in quickstart or registration mode.
Examples	mqttagwmode 1 mqttagwmode 2 <org_id> <registration token>

6.1.13 Load Default Command

loaddefault

loaddefault

Parameters	None
Description	This command is used to put the gateway board in default mode.
Examples	<ul style="list-style-type: none"> loaddefault

6.1.14 Trigger OTA Command

triggerota

triggerota

Parameters

None

Description

This command is used to start the OTA process of the gateway.

Examples

- triggerota

6.2 MQTT—Enabling IoT

MQTT protocol is a lightweight machine-to-machine connectivity protocol. MQTT protocol is based on the publish-and-subscribe messaging model and is designed to be used on top of TCP / IP protocol.

The key point of this protocol includes small code footprint and low network bandwidth requirements. Other features include a faster response time, low power requirement, and ease of scalability. All these advantages make it an ideal candidate for communication protocol in embedded devices intended to implement IoT (internet of things) applications. More information regarding MQTT protocol can be obtained from the latest MQTT protocol specification.

A simple MQTT infrastructure contains a broker (like a central hub) connected to multiple clients, each of which has the capability of publishing on any topic (token). The broker sends the message published on any topic to all the subscribers of that topic, as shown in Figure 12.

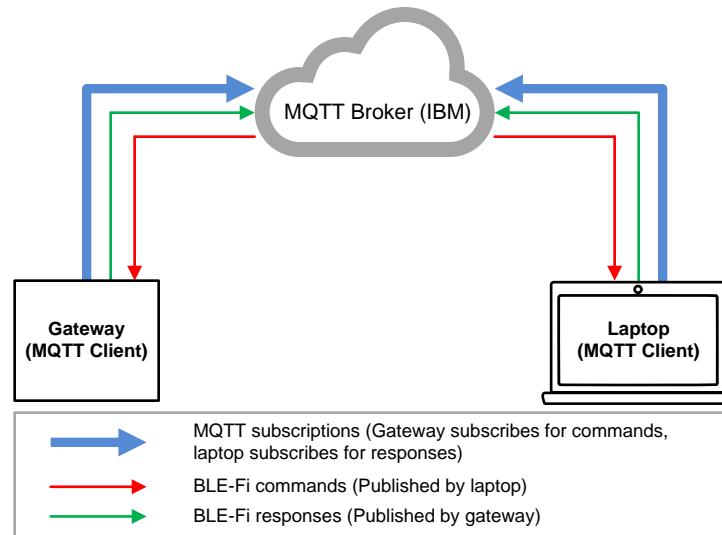


Figure 12. MQTT Connection

More detailed info on MQTT can be found in the following links:

- <http://mqtt.org/faq>
- http://processors.wiki.ti.com/index.php/CC32xx_MQTT_Client

6.2.1 Connection

The gateway must be connected to an AP (with Internet connection) for MQTT to work. Refer to [Section 5.3](#) for more details.

The gateway connects to the IBM MQTT server during initialization.

6.2.2 MQTT Clients

The gateway supports one MQTT client that can support up to 8 devices. This client can operate in either IBM Quickstart mode or Registered mode with the corresponding IBM servers.

6.2.3 MQTT Gateway Client

Quickstart mode is enabled when CC2650 SensorTags connect to the gateway clients are created.

This mode provides an intuitive feel of the entire platform, with richness of live sensor data of CC26xx SensorTags connected to the gateway, being streamed and displayed on the cloud from a remote location.

Procedure:

1. Connect the CC2650 SensorTags using CLI or MQTT (Gateway client—see).
2. Open the URL <http://quickstart.internetofthings.ibmcloud.com>.
3. Type the BD address of the CC26xx SensorTag as the device ID, which is scanned and displayed on the CLI.
4. Click the Go button.

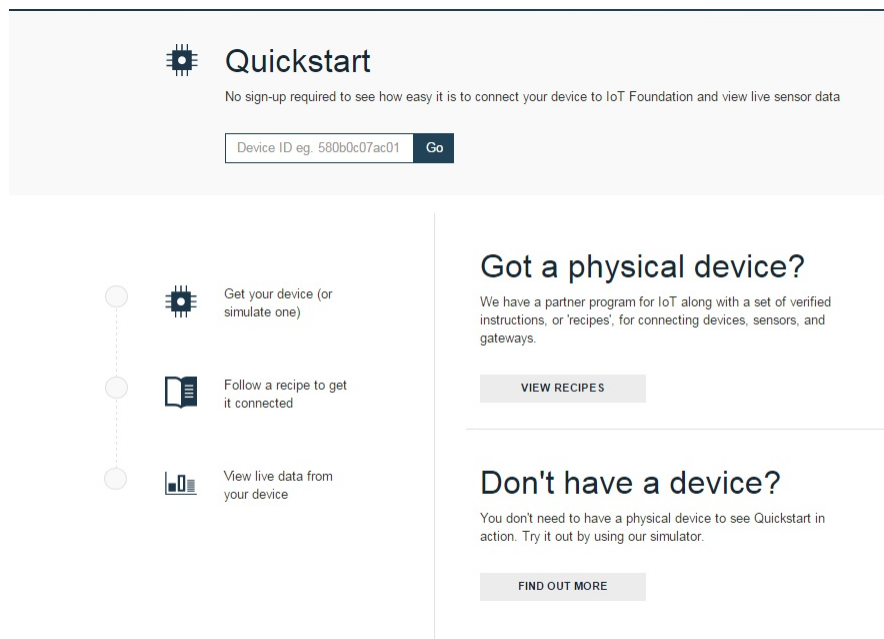


Figure 13. MQTT Device Quickstart

5. When connected, the device displays its status as *Connected* with the time (see [Figure 14](#)).

```

*****
BleFi - BLE-WiFi gateway <Texas Instruments>
Version : 1
*****

[L2NMP BLE Engine] BLE Engine initialized
[NPI] Waiting for SBL to be complete ...
[WLAN EVENT] Device Connected to the AP: MBR1400-1b3 , BSSID: 0:30:44:18:f1:b3

[WiFi] Connecting to network
[WiFi] Auto Connecting...[NETAPP EVENT] IP Acquired: IP=192.168.0.122 , Gateway
=192.168.0.1

[OTA] WLAN Connected
[OTA] Waiting for OTA trigger
[L2NMP BLE] SimpleLink Ready, starting BLE
[WiFi] Autoconnect Success
[MQTT] Configuration file Present
[L2NMP Core] Reading stored L2NMP configuration from database
G:>
[MQTT] GW Mode : registered
[MQTT] Gw Server : iigxjd.messaging.internetofthings.ibmcloud.com
[MQTT] Client ID d:iigxjd:gateway:5c313e0322d4
[MQTT] Server : iigxjd.messaging.internetofthings.ibmcloud.com
[MQTT] Client-Id : d:iigxjd:gateway:5c313e0322d4
[MQTT] Connecting to broker...
[SBL] Firmware Up to date
[L2NMP BLE] Waiting for NPI initialization ...
[L2NMP BLE] NPI initialized.
[L2NMP BLE] Starting the BLE Stack
[L2NMP BLE] BLE Stack Initialized
[MQTT] Broker connect Successful
G:> █
    
```

Figure 14. Device Connected

6. The device also displays live sensor data of the connected CC26xx SensorTag.

6.2.4 MQTT—Remote Access of the Gateway

MQTT works on a subscription-publish mechanism. The gateway, as an MQTT client subscribes to numerous topics. These subscribed topics act as a command to the gateway. A remote client may publish one of these topics with proper parameters as a command to the gateway. As a response to the command, the gateway publishes some topics. To receive the response, the remote client must have subscribed to these topics. Figure 15 depicts the working of MQTT on the gateway.

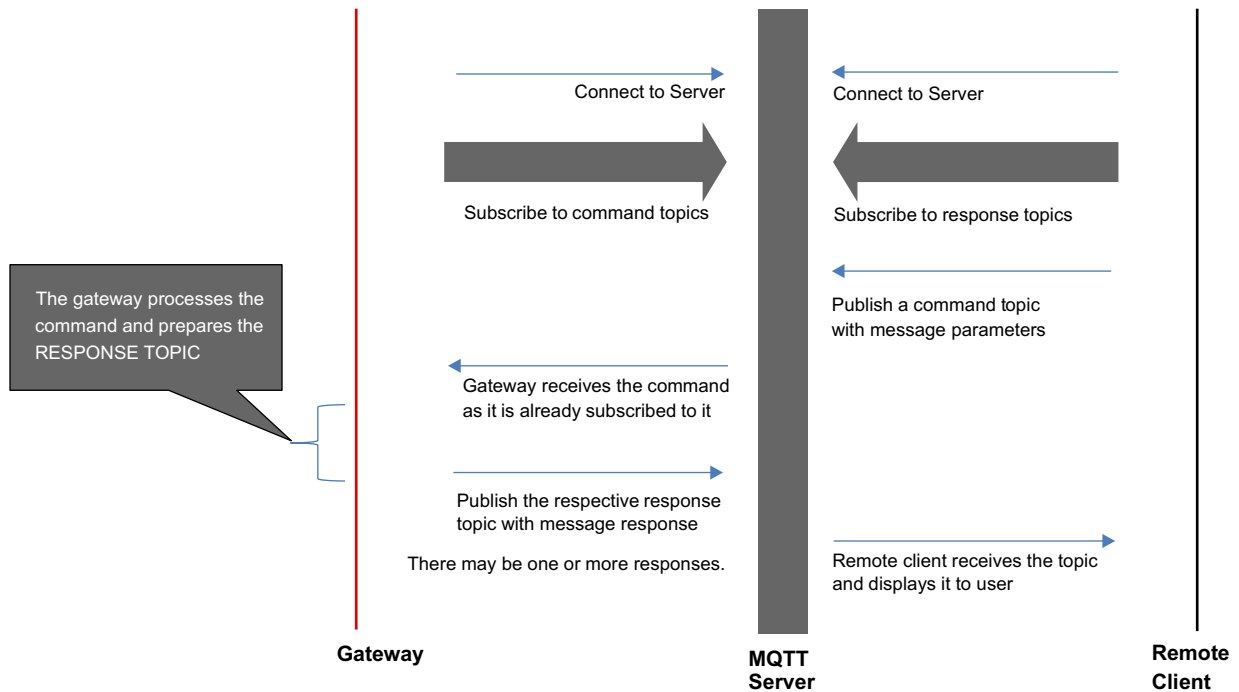


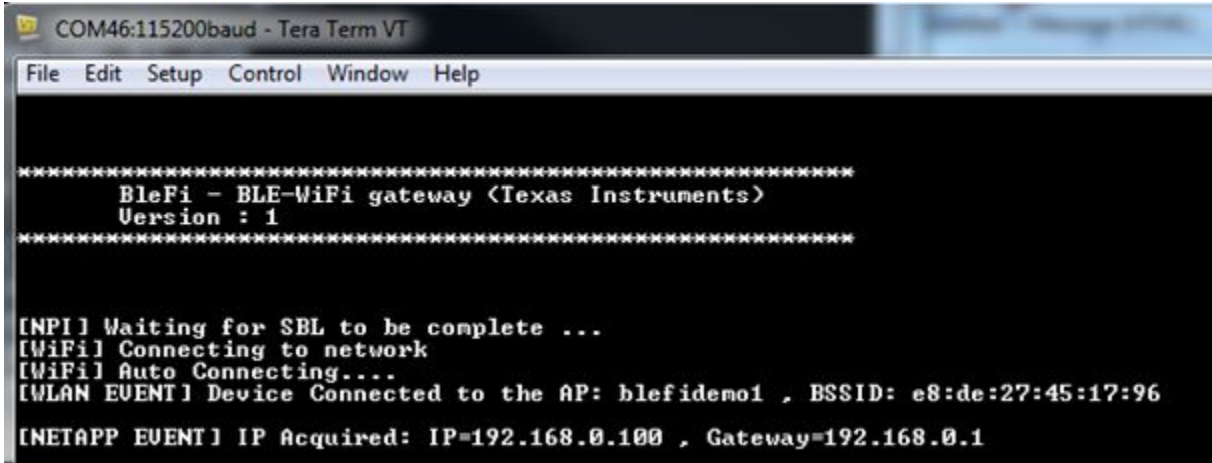
Figure 15. MQTT Sequence Diagram

The gateway maintains two different contexts while connecting to the MQTT server. From external client point of view, there is no difference while accessing the topics related to either of the contexts.

- **Gateway Context (1 number)**—This context is the default context and is created during initialization. gateway context contains the topics that are related to gateway and a *Bluetooth* Smart device.

6.3 Bluetooth Smart Dashboard

1. The HTML pages are part of the installer. These pages should be downloaded onto SFLASH of the gateway.
2. Run the gateway application. When connected to the AP, the acquired IP is displayed in the CLI as shown in [Figure 16](#).



```

COM46:115200baud - Tera Term VT
File Edit Setup Control Window Help
*****
BleFi - BLE-WiFi gateway (Texas Instruments)
Version : 1
*****

[INPI] Waiting for SBL to be complete ...
[WiFi] Connecting to network
[WiFi] Auto Connecting...
[WLAN EVENT] Device Connected to the AP: blefidemo1 , BSSID: e8:dc:27:45:17:96
[NETAPP EVENT] IP Acquired: IP=192.168.0.100 , Gateway=192.168.0.1
  
```

Figure 16. IP Address Displayed on CLI

3. Open the browser and type <ipaddr>/ble_dashboard.html
 - For example, type 192.168.1.100/ble_dashboard.html in browser URL field
 - Browser shows the *Bluetooth* Smart dashboard page

The *Bluetooth* Smart dashboard page contains two tabs:

- **Connection**—Users can scan all the advertising *Bluetooth* Smart devices in the gateway vicinity using this tab. This tab contains controls to establish and terminate link to a *Bluetooth* Smart device
- **Data**—This tab is used to show all the characteristics of a connected *Bluetooth* Smart device. Based on the permissions, the characteristic value can be read from or written to, using this tab

6.4 Bluetooth Smart Connection Page

Figure 17 shows *Bluetooth* Smart Dashboard Page.

The screenshot shows a web browser interface for the Bluetooth Smart Dashboard. At the top, there is a navigation menu with options: Welcome, Overview, About, Setup, **BLE**, Developer's Portal, and Demos. Below the menu, there are two tabs: 'Connection' and 'Data'. The main content area is divided into three sections:

- BLE Settings:** Includes a 'Start scanning for devices' label and a 'Scan' button.
- Scanned devices:** A table with columns: Scan ID, Dev name, BD addr, and Addr type. It contains one entry: Scan ID 1, Dev name SensorTag, BD addr 0xB4994C64C9B5, and Addr type 0. Below the table is an 'Establish Link' button.
- Connected devices:** A table with columns: Conn ID, Dev name, BD addr, and Addr type. It contains one entry: Conn ID 0, Dev name SensorTag, BD addr 0xBC6A29C35FE9, and Addr type 0. Below the table is a 'Terminate link' button. A 'Refresh' button is located to the right of the table header.

Figure 17. *Bluetooth* Smart Dashboard Page

6.4.1 Scanning a Bluetooth Smart Device

1. Select the Connection tab.
2. Press the *Scan* button to start scanning for available *Bluetooth* Smart devices.
3. Wait for 5 seconds for the command to be completed.
4. Press the Refresh button to show the list of scanned devices.

6.4.2 Establishing Link with a Bluetooth Smart Device

1. Among the list of scanned devices, Select the Radio Button of the desired *Bluetooth* Smart device.
2. Press the *Establish Link* button.
3. Wait for 5 seconds for the command to be completed.
4. Press the Refresh button to show the list of connected devices.

6.4.3 Terminating Link from a Bluetooth Smart Device

1. Among the list of connected devices, Select the Radio Button of the desired *Bluetooth* Smart device.
2. Press the *Terminate Link* button.
3. Wait for 2 seconds for the command to be completed.
4. Press the Refresh button to show the updated list of connected devices.

6.5 Bluetooth Smart Data Page

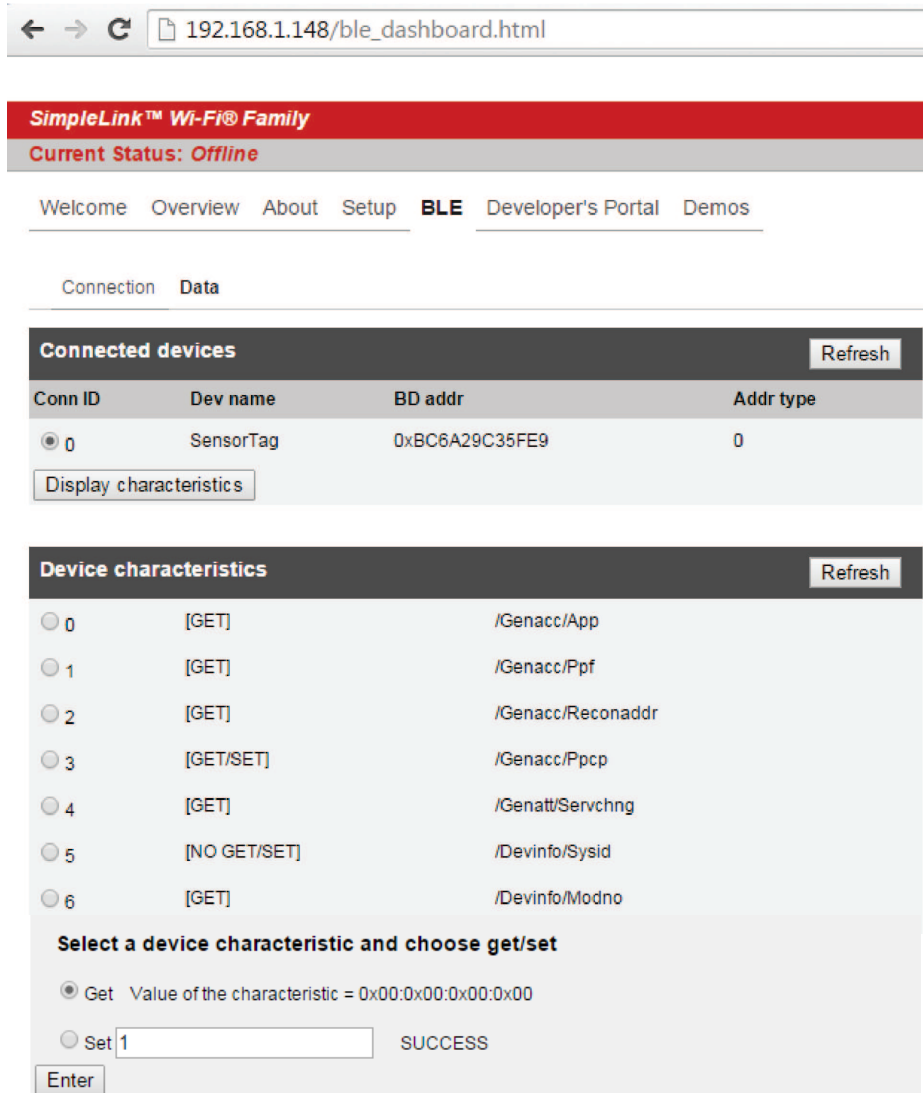


Figure 18. Bluetooth Smart Data Page

6.6 Display Characteristics of a Connected Bluetooth Smart Device

1. Select the Data tab.
2. Select the radio button of the desired *Bluetooth* Smart device from the Connected Devices table.
3. Press the Display Characteristics button to show the characteristics of the selected *Bluetooth* Smart device.

6.7 Read or Write Characteristic Value

1. In the Device Characteristics table, select the radio button of the characteristic to be read or written.
2. To read a characteristic, select the Get radio button and press the Enter button.
3. The characteristic value read will be displayed adjacent to the Get button.
4. To write to a characteristic value, select the Set radio button, enter the value to be written in the text box adjacent to the Set button, and press the Enter button.
5. If the write is completed successfully, the SUCCESS message is shown.

6.8 OTA

The OTA update is the wireless delivery of new software updates and/or configurations to embedded devices and with the concept of Wireless Sensor Network and Internet of Things, OTA is an efficient way of distributing firmware updates or upgrades.

The application binary (ugateway.bin) and the gateway enabler binary (cc26xx.bin) files can be updated OTA. This feature requires the user to create a Dropbox account and load the files in the account.

6.8.1 Creating a Dropbox API Application

1. Create an account with [Dropbox](#) and log in.
2. Go to <https://www.dropbox.com/developers/apps/create>
3. Choose *Dropbox API* app.
4. Choose *Files and Datastores* and *Yes, my app only needs access to files it creates*.
5. Provide a name for the application
6. Click the *Create APP* button.
7. Scroll down to *Generated access token* on the Apps setting page.
8. Click generate.
9. Copy and save the generated token (this token must be stored in the gateway using the *otaconfig* command in CLI).
10. Go to <https://www.dropbox.com/home/Apps>.
11. Click on the application name.
12. Create a new folder and name it in the following format: `TI_BleFi_v<version number>` (for example: `TI_BleFi_v01`). The version name is found on the CLI banner (see [Figure 19](#)).

```

*****
BleFi - BLE-WiFi gateway (Texas Instruments)
Version : 1
*****

```

Figure 19. CLI Banner Showing Version Number

13. Rename *blefi.bin* as *f80_sys_mcuimgA.bin*.
14. Rename *ble_gateway_enabler.bin* as *f80_cc26xx.bin*.
15. Copy these files to the new folder in the Dropbox account.
16. Power on the gateway.
17. Connect to an AP (with internet access) after initialization (*Waiting for OTA Trigger* on the CONSOLE indicates that the OTA can be performed. .
- 18.
19. Press the BTN-1 to start the OTA process.
20. When the OTA download is completed, the gateway restarts.

7 Test Data

This section describes the tests performed on the gateway.

7.1 CLI Snapshot

Figure 20 shows a snapshot of CLI.

```

*****
      BleFi - BLE-WiFi gateway <Texas Instruments>
      Version : 1
*****

[NPI] Waiting for SBL to be complete ...
[WiFi] Connecting to network
[WiFi] Auto Connecting....
[GW] SimpleLink Ready, starting Gateway
[WLAN EVENT] Device Connected to the AP: blefitata , BSSID: 1c:8e:5c:98:f2:2c

[NETAPP EVENT] IP Acquired: IP=192.168.1.102 , Gateway=192.168.1.1

[OTA] WLAN Connected
[OTA] Waiting for OTA trigger
[WiFi] Autoconnect Success
[GW] Device bonding information file <blefi.cfg> Present
Autoscan 60 seconds
[MQTT] GW Mode : demo
[MQTT] Gw Server : 192.84.45.44
[MQTT] Dev Mode : quickstart
[MQTT] Dev Server : quickstart.messaging.internetofthings.ibmcloud.com
[MQTT] Client ID f4b85e4576ad
[MQTT] Server : 192.84.45.44
[MQTT] Client-Id : f4b85e4576ad
[MQTT] Connecting to broker...
[SBL] No cc26xx.bin file to update
[SBL] Firmware Up to date
[GW] Waiting for NPI initialization ...
G:>
[GW] NPI initialized.
[GW] Starting the BLE Stack
[GW] BLE Stack Initialized

[MQTT] Broker connect Successful
[MQTT] Subscribing to topics...
[MQTT] Subscription to these topics successful
/blefi/find
/blefi/f4b85e4576ad/scan
/blefi/f4b85e4576ad/linke
/blefi/f4b85e4576ad/linkt
    
```

Figure 20. CLI Snapshot

7.2 Module Test Matrix

Table 5. Module Test Matrix

SL NUMBER	TESTS	RESULT	COMMENTS
1	Endurance Test	PASS	48 hours, 8 sensor tags connected. MQTT is enabled in both quickstart mode with no TLS security and registered mode
2			Gateway Quickstart and registered mode
3	CLI	PASS	All commands
4	Gateway	PASS	Data bridging between WLAN and <i>Bluetooth</i> Smart
5	<i>Bluetooth</i> Smart Central	PASS	GAP and GATT commands
6	Wi-Fi Configuration	PASS	Connecting and disconnecting from AP
7	<i>Bluetooth</i> Smart Connections	PASS	Up to three 2650 sensor tags.
8	OTA Update	PASS	Update Gateway Binary and CC26xx Binary
9	OS Functionality	PASS	TI-RTOS

8 Design Files

8.1 Schematics

To download the schematics, see the design files at [TIDC-BLE-TO-WIFI-IOT-GATEWAY](#).

8.2 Bill of Materials

To download the bill of materials (BOM), see the design files at [TIDC-BLE-TO-WIFI-IOT-GATEWAY](#).

8.3 Layer Plots

To download the layer plots, see the design files at [TIDC-BLE-TO-WIFI-IOT-GATEWAY](#).

8.4 Altium Project

To download the Altium project files, see the design files at [TIDC-BLE-TO-WIFI-IOT-GATEWAY](#).

8.5 Layout Guidelines

To download the layout guidelines, see the design files at [TIDC-BLE-TO-WIFI-IOT-GATEWAY](#).

8.6 Gerber Files

To download the Gerber files, see the design files at [TIDC-BLE-TO-WIFI-IOT-GATEWAY](#).

8.7 Assembly Drawings

To download the assembly drawings, see the design files at [TIDC-BLE-TO-WIFI-IOT-GATEWAY](#).

8.8 Software Files

To download the software files, see the design files at [TIDC-BLE-TO-WIFI-IOT-GATEWAY](#).

9 References

1. *CC3200 SimpleLink™ Wi-Fi® and Internet-of-Things solution, a Single-Chip Wireless MCU (CC3200)*
2. *SimpleLink™ multi-standard 2.4 GHz ultra-low power wireless MCU (CC2650)*
3. *Ultralow-Noise, High PSRR, Fast, RF, 1-A Low-Dropout Linear Regulators (TPS79601)*
4. *2-Channel ESD Solution for SuperSpeed USB 3.0 Interface (TPD2EUSB30)*
5. Silabs: *USB to UART Bridge* (<http://www.silabs.com/products/interface/usbtouart/Pages/usb-to-uart-bridge.aspx>)

10 About the Authors

SREEHARSHA SRINIVAS Applications Manager, LPRF, Texas Instruments.

VIJAYSARATHY SHASTRI Applications Lead, LPRF, Texas Instruments.

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from B Revision (January 2016) to C Revision	Page
• Removed "HTTP server and pages" and "Gateway module (central and database)" and replaced with "L2NMP_COMMON (generic interface with upper layer agent and CLI, L2NMP authentication and database)" and "L2NMP_PSS (BLE) (central and BLE database)" from module list.....	7
• Removed HTTP references throughout document	9
• Removed "The gateway can be controlled over a local LAN using HTTP pages." following "The primary feature of the gateway is to make the characteristics of a connected <i>Bluetooth</i> Smart device available on the cloud (IoT using MQTT).".....	9
• Changed "gateway" to "ugateway" throughout document.	13
• Removed "blefi" and "schema" from step 4 of Source Build, changed gateway to "ugateway", added "l2nmp" and "l2nmp_ble".	14
• Removed projects from step 5 and changed to "Build clean all the previously mentioned projects."	14
• Changed "blefi.bin" to "ugateway.bin".....	14
• Removed note that said "Only Just Works bonding is supported, and no Passkey support is present while connecting to <i>Bluetooth</i> Smart devices." and "The CLI commands to connect to a <i>Bluetooth</i> Smart device are scan and linke. Scan is used to scan the devices that are advertising in the vicinity of BleFi. Linke connects to a desired device. The parameter of the linke command is the <scan id>." and "For example, G:>scan..G:>linke 0".....	17
• Removed HTTP column from table 4. Removed "Remove Pairing Information" row. Added "Configure Max Devices", configure gw in proxy or cache (HL confirmation), configure attachment mode (all, enabled, authorized), manage the list of authorized devices (manage db), Retrieve Wi-Fi of GW".....	19
• Replaced "servername" with "<optional fields>" after mqttmode in synopsis description.	23
• Replaced "0 for demo mode" with "2 for registration mode" in after [mode] and removed "[servername] - url or ip address of the server" in Parameters.....	23
• Changed "Demo" to "registration" after "reset the MQTT mode in quickstart or" and removed "In Demo mode, a server name needs to be specified." in Description.	23
• Replaced "mqttgwmode 0 "192.84.45.44"" with "mqttgwmode 1, mqttmode 2 <org_id>, <registration_token>" in Examples.....	23
• Removed "MQTT Device Mode Command" section.	23
• Removed "Auto Scan Command" section, "Display Auto-Connect List Command" section, "Remove Auto Connection Command" section, and "Auto-Connect Command" section.	24
• Removed "These prints on the CONSOLE (CLI) confirm the connection." and "If these prints do not appear on CONSOLE, one of the following issues may have occurred: Gateway-AP connection is not successful, AP is not connected to internet, and IBM MQTT server is not responding" and note that said "The 5c313e032287 is the MAC address of the gateway. This address is just an example and will be different for different gateway devices." and an example image, after "The gateway connects to the IBM MQTT server during initialization."	25
• Changed "two kinds of MQTT clients: a gateway client and device clients. The gateway client is created when the gateway powers up. By default, the gateway client is connected to an MQTT demo server of IBM. The device clients are created when a CC2650 SensorTag is connected to the gateway (over <i>Bluetooth</i> Smart). There can be a maximum of 3 CC2650 SensorTag device clients. By default, the CC2650 SensorTag device clients are connected to the QuickStart server of IBM." to "one MQTT client that can support up to 8 devices. This client can operate in either IBM Quickstart mode or Registered mode with the corresponding IBM servers."	25
• Changed "MQTT Device Client" to "MQTT GW Client" and removed "and device" after "connect to the gateway".....	25
• Removed "Device Context (0 to 3 numbers)—A device context is created when a <i>Bluetooth</i> Smart device gets connected to the gateway. The maximum number of device contexts that can be created in the gateway is three." after Gateway Context (1 number) bullet point.	27
• Removed 6.2.6 Command Topics (Available Only in Demo Mode) section, 6.2.7 Response Topics (Available Only in Demo Mode) and 6.3 HTTP section.	27

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ("TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products (<http://www.ti.com/sc/docs/stdterms.htm>), [evaluation modules](#), and [samples](http://www.ti.com/sc/docs/sampterm.htm) (<http://www.ti.com/sc/docs/sampterm.htm>).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2018, Texas Instruments Incorporated