CC2640R2F BLUETOOTH MCU

INTRODUCTION

The CC2640R2F device is a wireless microcontroller (MCU) targeting Bluetooth® 4.2 and Bluetooth 5 low-energy applications.

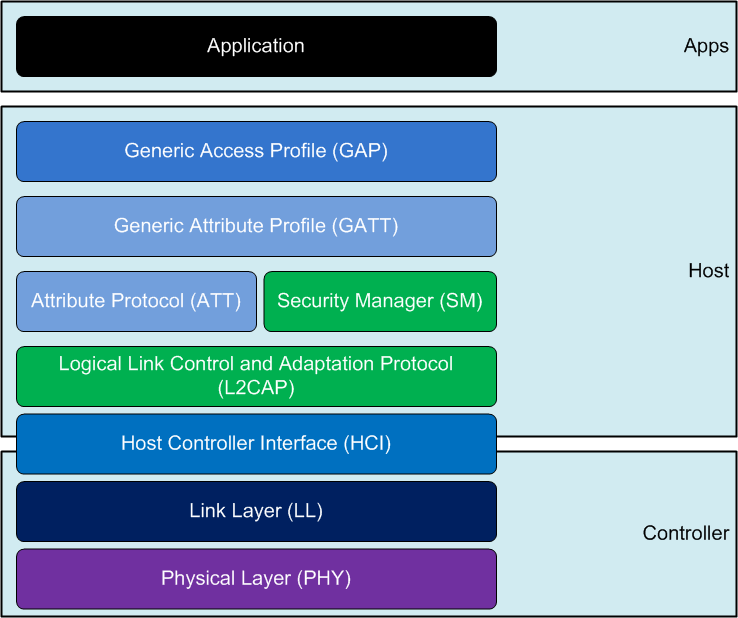
The device is a member of the SimpleLink™ ultra-low power CC26xx family of cost-effective, 2.4-GHz RF devices. Very low active RF and MCU current and low-power mode current consumption provide excellent battery lifetime and allow for operation on small coin cell batteries and in energy-harvesting applications.

The SimpleLink Bluetooth low energy CC2640R2F device contains a 32-bit ARM® Cortex®-M3 core that runs at 48 MHz as the main processor and a rich peripheral feature set that includes a unique ultra-low power sensor controller. This sensor controller is ideal for interfacing external sensors and for collecting analog and digital data autonomously while the rest of the system is in sleep mode. Thus, the CC2640R2F device is great for a wide range of applications where long battery lifetime, small form factor, and ease of use is important.

BASICS – (Reference: SimpleLink Academy)

FUNDAMENTALS OF BLUETOOTH 4.2:

The Generic Access Profile (GAP) is a top layer in the host protocol stack that defines how BLE devices behave in standby and connecting states to maintain interoperability with peer devices. GAP also describes discovery, link establishment and security procedures. The GAP APIs can be called from the application layer.



## **GATT (Services and Characteristics)**

For Bluetooth low energy, communication occurs over the air according to the Attribute Protocol (ATT). From a BLE application point of view however, data is exchanged using the Generic Attribute Protocol (GATT) which can be viewed as a meta-layer on top of ATT. [Bluetooth SIG](https://www.bluetooth.com/) has defined several Profiles for the use of these protocols to ensure interoperability.

A Profile is a written document (or a cryptic voicemail) describing how a number of GATT Services and GATT Characteristics (defined separately from the Profile) should be used to achieve a certain application

The Generic Attribute Profile (GATT) establishes in detail how to exchange all profile and user data over a BLE connection. In contrast with GAP, which defines the low-level interactions with devices, GATT deals only with actual data transfer procedures and formats. GATT uses the Attribute Protocol (detailed in [Attribute Protocol (ATT)](https://www.oreilly.com/library/view/getting-started-with/9781491900550/ch02.html#att)) as its transport protocol to exchange data between devices. This data is organized hierarchically in sections called services, which group conceptually related pieces of user data called characteristics.

A universally unique identifier (UUID) is a 128-bit (16 bytes) number that is guaranteed (or has a high probability) to be globally unique. UUIDs are used in many protocols. Attributes are the smallest data entity defined by GATT (and ATT). They are addressable pieces of information that can contain relevant user data (or metadata) about the structure and grouping of the different attributes contained within the server. The attribute handle is a unique 16-bit identifier for each attribute on a particular GATT server. It is the part of each attribute that makes it addressable, and it is guaranteed not to between transactions or, for bonded devices, even across connections. Because value 0x0000 denotes an invalid handle, the amount of handles available to every GATT server is 0xFFFE (65535), although in practice, the number of attributes in a server is typically closer to a few dozen.

The attribute type is nothing other than a UUID .This can be a 16-, 32-, or 128-bit UUID, taking up 2, 4, or 16 bytes, respectively. The type determines the kind of data present in the value of the attribute, and mechanisms are available to discover attributes based exclusively on their type. Permissions are metadata that specify which ATT operations can be executed on each particular attribute and with which specific security requirements. The attribute value holds the actual data content of the attribute. There are no restrictions on the type of data it can contain (you can imagine it as a non-typed buffer that can be cast to whatever the actual type is, based on the attribute type), although its maximum length is limited to 512 bytes by the specification.

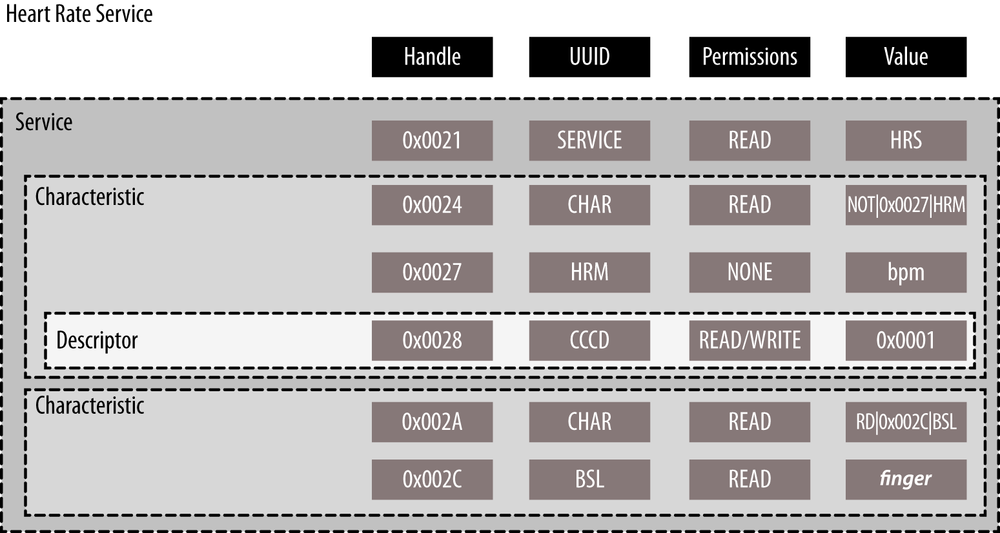
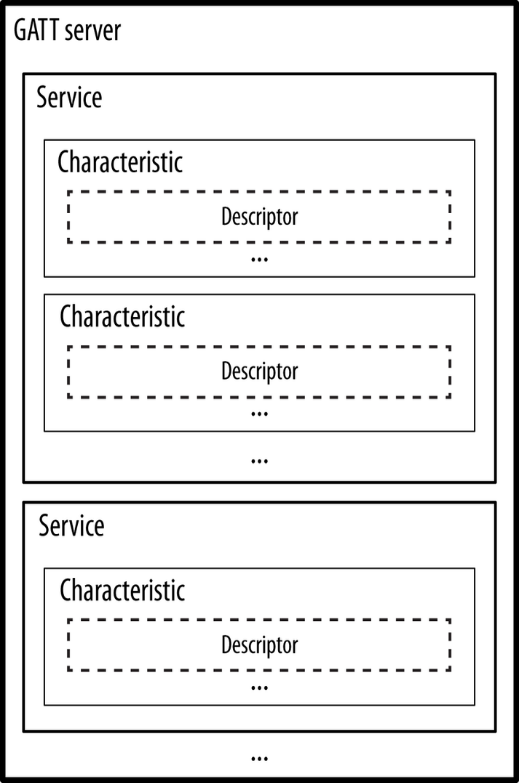
### Client Characteristic Configuration Descriptor- The name of this descriptor type is perhaps not very poetic. Often abbreviated to CCCD, the descriptor is an attribute with the UUID 0x2902 and is readable and writable.The value a GATT Client writes to this attribute will determine whether the GATT Server is allowed to send Notifications (if 0x0001 is written) or Indications (if 0x0002 is written).

### Services

GATT services group conceptually related attributes in one common section of the attribute information set in the GATT server. The specification refers to all the attributes within a single service as the service definition. Therefore, a GATT server’s attributes are in fact a succession of service definitions, each one starting with a single attribute that marks the beginning of a service (aptly named a service declaration.)

### Characteristics

You can understand characteristics as containers for user data. They always include at least two attributes: the characteristic declaration (which provides metadata about the actual user data) and the characteristic value (which is a full attribute that contains the user data in its value field).



Above example on the right side describes HEART RATE SERVICE.

**Advertising**

Bluetooth devices send advertising packets (PDUs) to broadcast data, and to allow other devices (scanners) to find and connect to them. The advertising data consists up to 31 bytes of user configurable data. An additional 31 bytes can be sent as a scan response to a scan request.

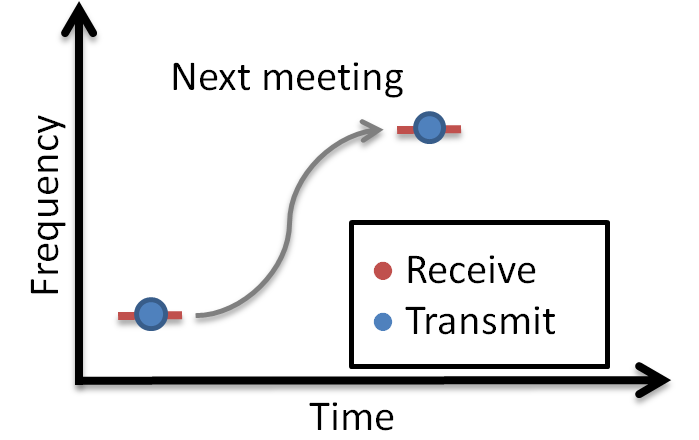
# Scanning

When not connected, Bluetooth devices can either advertise their presence by transmitting advertisement packets or scan for nearby devices that are advertising. This process of scanning for devices is called device discovery. There are two types of scanning; active and passive. The difference is that an active scanner can send a scan request to request additional information from the advertiser, while a passive scanner can only receive data from advertising device. Note that the terms discovery and scanning may be used interchangeably.

# About BLE Connections

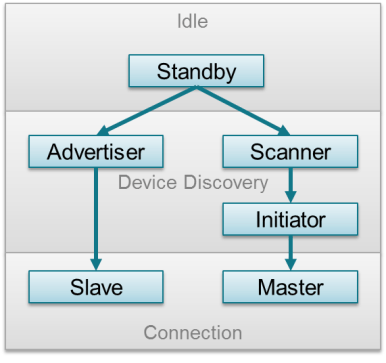
What is a BLE connection? A connection implies a link between devices over time. BLE is a synchronous radio frequency (RF) protocol, meaning that any transmission between devices must be scheduled. A BLE connection can thus be perceived as a series of meetings where two devices transmit and receive information at the same time, on the same radio frequency. In order for this to work, the devices must agree on where (that is, on what frequency) and when next to meet. The BLE-Stack handles connection timing and frequency hopping.

A BLE connection always consists of two devices. The device that initiates the connection is called the master. This device also has the final word on timing and frequency hopping. The other device is called the slave.



## The Link Layer

The Link Layer is a part of a Bluetooth Low Energy application that keeps track of whether the device is in a connected state (link) or not. In order for BLE devices to find each other, they must either scan or advertise. A scanning device that has found a connectable advertiser can initiate a connection.



A peripheral is device with the ability to advertise, and enter a connection as the slave. A central is a device that can scan for BLE devices and initiate connections. The central will always be the master in the connection.