Application Report
SWRA475–January 2015

Bluetooth® Low Energy Beacons

Joakim Lindh

ABSTRACT

This application report presents the concept of beacons by using *Bluetooth* low energy technology. The important parameters when designing beacon solutions are elaborated on a detailed level throughout the document. With the use of the TI BLE-Stack, beacons can be done in a simple and intuitive way.

Contents

List of Figures

List of Tables

1 What is a Beacon?

Beacon in wireless technology is the concept of broadcasting small pieces of information. The information may be anything, ranging from ambient data (temperature, air pressure, humidity, and so forth) to micro-location data (asset tracking, retail, and so forth) or orientation data (acceleration, rotation, and so forth).

The transmitted data is typically static but can also be dynamic and change over time. With the use of *Bluetooth* low energy, beacons can be designed to run for years on a single coin cell battery. This application report introduces the concept of beacons and how to get started with implementing a beacon solution. Naming conventions throughout this document can be summarized as Beacons that broadcast information by using advertisements with *Bluetooth* low energy technology that could be branded as *Bluetooth* Smart.

2 *Bluetooth* Low Energy and *Bluetooth* Smart

A *Bluetooth* low energy device can operate in four different device roles. Depending on the role, the devices behave differently. The first two roles are connection-based:

- A Peripheral device is an advertiser that is connectable and can operate as a slave in a connection. This could, for example, be a health thermometer or a heart rate sensor.
- A Central device scans for advertisers and can initiate connections. It operates as a master in one or more connections. Good examples are Smartphones and computers.

This means that the device roles used for established connections are the Peripheral and the Central roles. The other two device roles are used for one-directional communication:

- A Broadcaster is a non-connectable advertiser, for example, a temperature sensor that broadcasts the current temperature, or an electronic tag for asset tracking.
- An Observer scans for advertisements, but cannot initiate connections. This could be a remote display that receives the temperature data and presents it, or tracking the electronic tag.

The two obvious device roles for beacon applications are Peripheral and Broadcaster. Both of them send the same type of advertisements with the exception of one specific flag that indicates if it is connectable or non-connectable. A Peripheral device that implements a GATT Server (GATT is an architecture for how data is stored and exchanged between two or more devices) can be branded as a *Bluetooth* Smart device. So a *Bluetooth* Smart branding indicates that the device is a connectable Peripheral device that has data, which could be interacted with.

A *Bluetooth* low energy solution is ideal for beacons because it is low power and the eco-system is already deployed in the majority of smartphones or other *Bluetooth* Smart Ready enabled devices on the market. The low-power consumption is achieved by keeping the transmission time as short as possible and allowing the device to go into sleep mode between the transmissions.

2.1 Non-Connectable Beacons

The non-connectable beacon is a *Bluetooth* low energy device in broadcasting mode. It simply transmits information that is stored internally. Because the non-connectable broadcasting does not activate any receiving capabilities, it achieves the lowest possible power consumption by simply waking up, transmit data and going back to sleep. This comes with the drawback of dynamic data being restricted to what is only known to the device, or data being available through external input from example serial protocols (universal asynchronous receiver/transmitter (UART), serial peripheral interface (SPI), universal serial bus (USB), and so forth).

2.2 Connectable Beacons

The connectable beacon is a *Bluetooth* low energy device in peripheral mode, which means that it cannot only transmit, but also receive as well. This allows a central device (for example, a smartphone) to connect and interact with services implemented on the beacon device. Services provide one or more characteristics that could be modified by a peer device. One example of these characteristic could be a string of data that represents the broadcasted information. This way, it is possible to have a configurable Beacon that is easily updated over the air.

2.3 Data Packet

The transmitted data from a *Bluetooth* low energy device is formatted according to the Bluetooth Core Specification and is comprised by the parts shown in Figure 1.

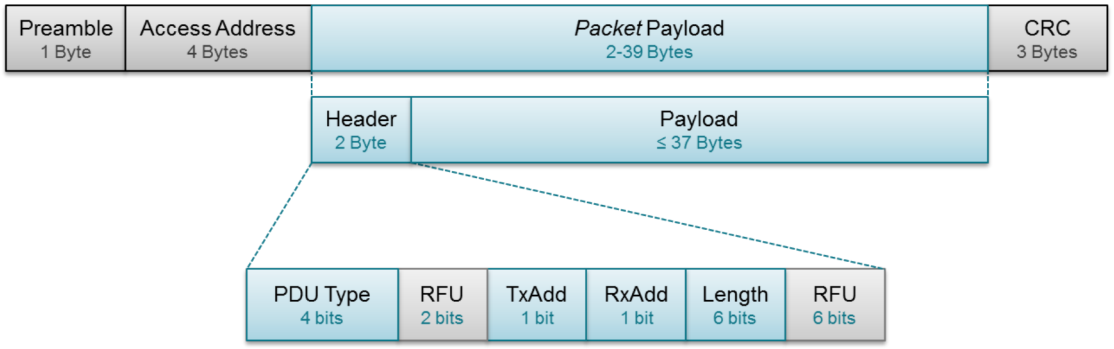

Figure 1. *Bluetooth* **Low Energy Data Packet**

The Preamble is a 1 byte value used for synchronization and timing estimation at the receiver. It will always be 0xAA for broadcasted packets. The Access Address is also fixed for broadcasted packets, set to 0x8E89BED6. The packet payload consists of a header and payload. The header describes the packet type and the PDU Type defines the purpose of the device. For broadcasting applications, there are three different PDU Types, as shown in Table 1. ADV_IND and ADV_NONCONN_IND have been described previously (as connectable and non-connectable) while ADV_SCAN_IND is simply a non-connectable broadcaster that can provide additional information by scan responses.

Table 1. Advertising PDU Types for Broadcasting Data

PDU Type	Packet Name	Description
0000	ADV_IND	Connectable undirected advertising event
0010	ADV_NONCONN_IND	Non-connectable undirected advertising event
0110	ADV_SCAN_IND	Scannable undirected advertising event

The TxAdd bit indicates whether the advertisers address (contained in the Payload) is public (TxAdd = 0) or random (TxAdd = 1). RxAdd is reserved for other types of packets not covered in this application note, as it does not apply to beacons.

The final part of the transmitted packet is the Cyclic Redundancy Check (CRC). CRC is an error-detecting code used to validate the packet for unwanted alterations. It ensures data integrity for all transmitted packets over the air.

The Payload of the packet includes the advertisers address along with the user defined advertised data as shown in Figure 2. These fields represent the beacons broadcasted address and data.

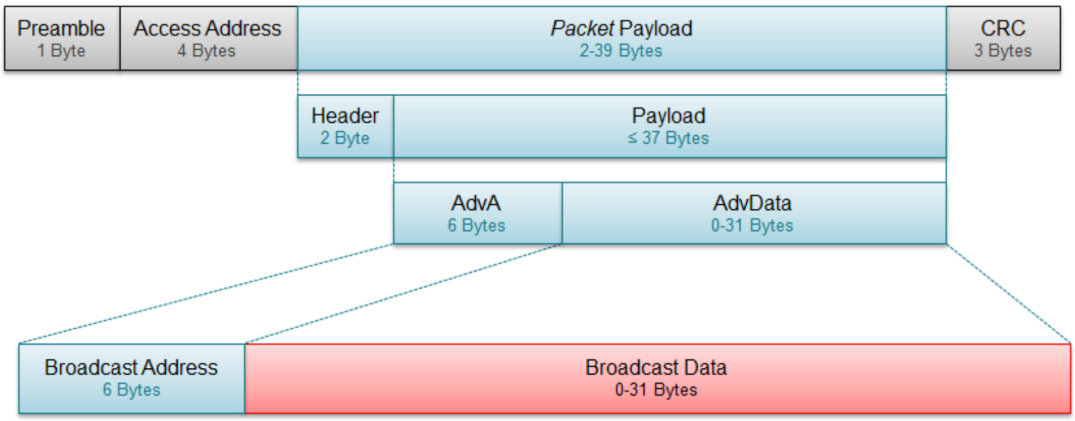

Figure 2. *Bluetooth* **Low Energy Broadcast Data**

2.4 Device Address

The broadcast address can be either public or random. A public address [1] (Vol.6.C.1.3 page 2500) is an IEEE 802-2001 standard and uses an Organizationally Unique Identifier (OUI) obtained from the IEEE Registration Authority. Texas Instruments provide IEEE addresses for all *Bluetooth* Smart devices. Random addresses can be directly generated by the beacon and can be of three different types: static, non-resolvable private and resolvable private. Static address is not allowed to be changed unless the device power cycles. A private address can change over time and a resolvable address can be used to derive the true address. A non-resolvable address can also change over time, which is the differentiation compared to a static address. The random address is a privacy feature that prevents tracking of a specific device. There are specific rules when generating random addresses and the details can be found in the Core Specification [1] (Vol3.C.10.8 page 2020).

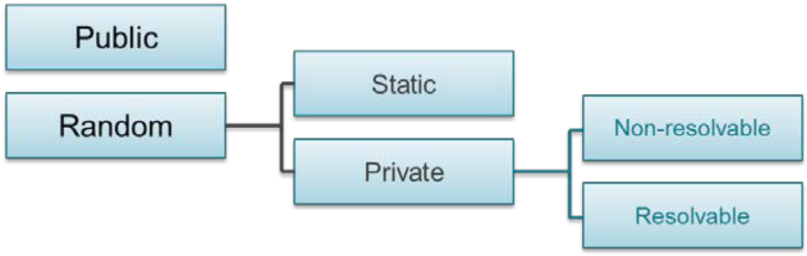

Figure 3. Beacon Address Types

The broadcasted data can be formatted according to *Bluetooth* SIG specified data formats, some examples shown in Table 2 [2], [3]. For the purpose of this application report, focus will be on the Flags and Manufacturer-Specific Data.

Table 2. Advertisement Data Types

AD Data Type	Data Type Value	Description
Flags	0x01	Device discovery capabilities
Service UUID	0x02 - 0x07	Device GATT services
Local Name	0x08 - 0x09	Device name
TX Power Level	0x0A	Device output power
Manufacturer Specific Data	0xFF	User defined

2.4.1 Flags

The first three bytes of the broadcasted data defines the capabilities of the device. It is a requirement from the Core Specification [1] (Vol 3.C.13.1.1 page 2029) and the format is defined as shown in Table 3.

Table 3. Advertisement Data Type, Flags

Byte	Bit	Flag/Value	Description
0		0x02	Length of this data
1		0x01	GAP AD Type Flags
2	0	LE Limited Discoverable Mode	180 s advertising
	1	LE General Discoverable Mode	Indefinite advertising time
	2	BR/EDR Not Supported	
	3	Simultaneous LE and BR/EDR (Controller)	
	4	Simultaneous LE and BR/EDR (Host)	
	5-7		Reserved

Discovery mode flags are bit masked and the various settings are presented in Table 3. If no bit flags are to be set, the Flags Data type can be omitted [2]. It would for example not be required for a non-connectable advertisement packet (ADV_NONCONN_IND).

2.4.2 Manufacturer Specific Data

When using Manufacturer-Specific Data, the 0xFF flag is used to indicate so. The first two bytes of the data itself should be a company identifier code.

Table 4. Advertisement Data Types, Manufacturer Specific Data Format

Byte	Value	Description
0	0x03-0x1F	Length of this data
1	0xFF	Manufacturer-Specific Data Flag
2	0x0D	Company ID
3	0x00	(Example, 0x000D – Texas Instruments)
4 - 31	-	User Defined Data (optional)

The *Bluetooth* low energy packet format allows a device to broadcast 25 Bytes of Manufacturer-Specific Data if the advertisement is of the type Connectable undirected advertising (ADV_IND) or Scannable undirected advertising event (ADV_SCAN_IND) as Discoverable Mode flags are required. For non-connectable undirected advertising (ADV_NONCONN_IND) maximum length of the Manufacturer-Specific Data is 28 bytes. The Manufacturer-Specific Data can contain any user-defined information.

The broadcasted data can also be formatted in a standardized way. At the time of writing this application report there are a couple of standards, such as iBeacon from Apple and AltBeacon, from Radius Networks. iBeacon is protected by the MFi license and is interoperable with all iOS devices. AltBeacon is an open standard and the specification can be downloaded at http://altbeacon.org/.

2.5 Broadcast Interval

A beacon achieves low-power consumption by residing in sleep most of the operating time, only waking up briefly to broadcast data. The time between these broadcast events are referred to as advertising interval, which is illustrated in Figure 4. For non-connectable beacons, the interval cannot be smaller than 100 ms. For connectable beacons, it cannot be smaller than 20 ms. To this interval, a 0-10 ms pseudo-random delay is added to ensure that beacons can coexist, even if they might start broadcasting at the same time.

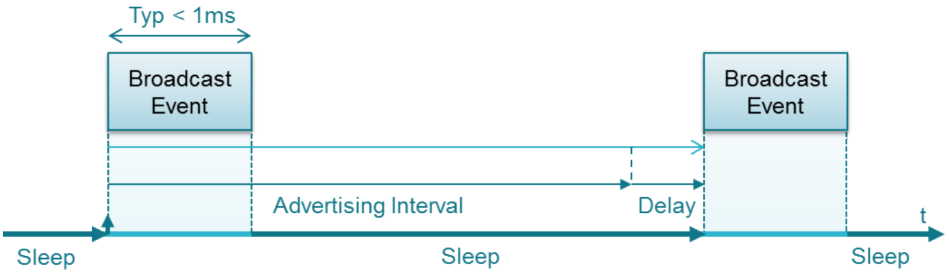

Figure 4. Advertising Interval

The advertising interval is chosen based on requirements of power consumption vs latency. Higher interval allows more time in sleep mode but increases the time it might take for an observer to obtain a broadcasted packet.

An observer typically uses a scanning with a duty cycles less than 100% to conserve energy or allow other wireless protocols a time slot. One good example is Smartphones, which in most cases has a one chip solution for *Bluetooth* and WiFi®. If an audio headset is connected via *Bluetooth* and WiFi has a connection to a local access point, *Bluetooth* low energy scanning will probably have a very low-duty cycle. The time slots are essentially split between the 2.4 GHz protocols on the device.

An observer can be either scanning in passive or active mode. If active mode is used and a beacon supports it, a Scan Request will be sent, which the beacon must respond to with a Scan Response. The request itself is an empty packet (no data allowed), whereas, the response is typically static information such as name or model of device. It is fully proprietary so it could also be calibration data for sensors or any other useful information. When an observer is scanning in passive mode, it will not send a Scan Request.

2.6 Power

A beacon can be powered using several methods. There are three different main approaches:

- DC power supply (USB, Mains, and so forth)
- Batteries (CR2032, AAA, Lithium, and so forth)
- Energy harvesting (Solar, kinetics, and so forth)

Typically, the primary choice is batteries that allow small and low-cost product designs while still maintaining a lifetime suitable for most applications. It is also possible to use re-chargeable batteries and in some applications together with wireless charging. The choice of battery type is important because some batteries might be sensitive to high peak currents. The amount of capacity needed is based on how often broadcasting is necessary, and if any further processing is required (sensor interaction, algorithms, and so forth). Sensor interaction typically involves serial communication using UART, SPI or inter-integrated circuit (I2C) that adds a power consumption overhead, which may be greater than the *Bluetooth* low energy protocol alone.

Figure 5. Power Options

If the design is powered by DC power supply, the assumption would be that the power consumption is no critical parameter. If it is still critical, the approach would be the same as for a battery powered design.

Energy harvesting is being introduced more into low-power wireless solutions and a beacon can be powered by energy harvested sources. Mechanical pressure, fresh fruit and solar power are commonly known sources and even indoor light [4] can be used to power a beacon.

2.7 Range

In Radio Frequency (RF) theory, the range is given by a number of factors:

- Sensitivity of the radio receiver
- Output power of the radio transmitter
- Environment and coexistence
- Antenna performance

A beacon application may require ranges from centimeters up several hundred meters. The maximum output power allowed by the core specification is 10 dBm, which should get distances up to several hundred meters if all above mentioned factors are considered. The mathematical and physical derivation of the theoretical range is not covered by this application report.

2.8 Coexistence

The open 2.4 GHz ISM frequency band that is used by *Bluetooth* low energy is filled with many other wireless protocols, such as Wi-Fi, as well as potential interference from home appliances, such as microwave ovens. These radio activities may interfere with the *Bluetooth* low energy activity. The broadcasting of advertisements occurs on three different channels sequentially as demonstrated in Figure 6.

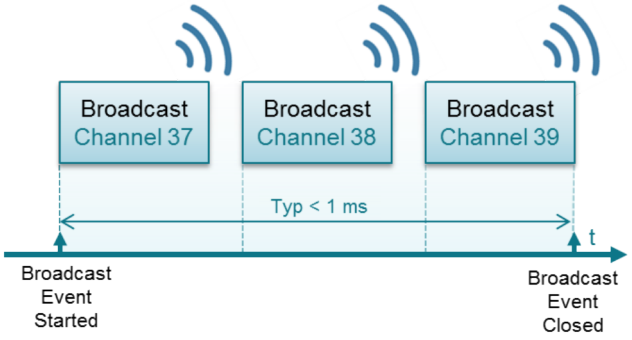

Figure 6. Broadcasting Advertisements

The channels 37, 38 and 39 have been chosen to not collide with the three most commonly used Wi-Fi channels; 1, 6 and 11, as shown in Figure 7.

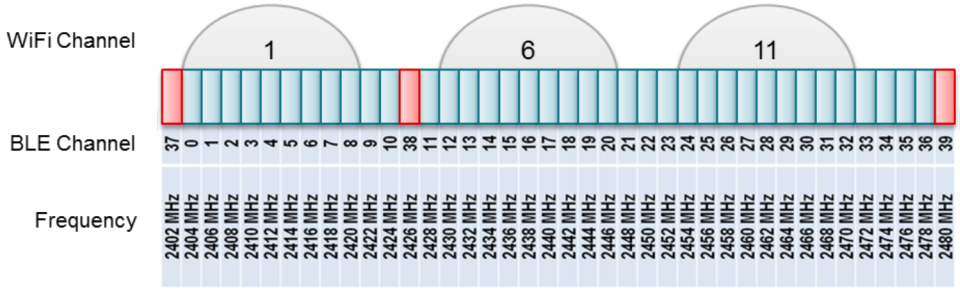

Figure 7. Frequency Band and Channels, *Bluetooth* Low Energy

However, Wi-Fi has significantly higher output power, up to 23 dBm compared to maximum allowed 10 dBm for *Bluetooth* low energy. This means that placing a beacon very close to a Wi-Fi source will probably distort the transmitted data as spurious emissions on side channels of the Wi-Fi unit will almost always occur on a non-ideal RF product, as show in Figure 8. Figure 8 shows the three broadcast output power peaks on advertising channels 37, 38 and 39 versus the Wi-Fi output peak for streaming data at 24 Mbit/s.

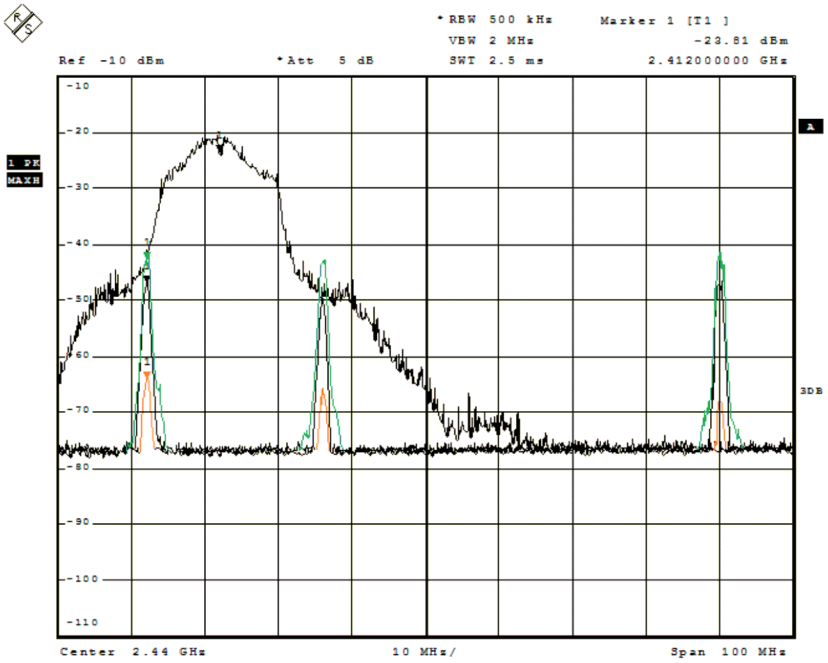

Figure 8. Spectrum for Wi-Fi Channel 1 versus Beacon Advertisements Channels

If Wi-Fi runs on other channels that overlap with the broadcasting channels, it is dependent on the received signal strength (RSSI) at the receiving device. Even though the advertising channels for *Bluetooth* low energy has been strategically placed in the 2.4 GHz ISM band to not interfere with the most common Wi-Fi channels, Figure 8 shows that interference and coexistence issues with these particular Wi-Fi channels could be present anyway. However, it should be noted that the frequency spectrum was measured with a beacon residing directly on top of a WiFi device.

Depending on the Beacon application, there are different requirements on coexistence. Even a small amount of active beacons tend to interfere with each other and cause packets to be lost. As already seen, Wi-Fi has in general higher TX-power as well as wider occupancy in the 2.4 GHz ISM band than *Bluetooth* low energy devices.

3 Designing a *Bluetooth* Low Energy Beacon

As with any design, there are always parameters that can be optimized to a certain cost, by weighting the different trade-offs. One example is the broadcasting interval. Choosing a lower interval increases the probability to be observed by a peer device quicker, although the power consumption is also increased.

3.1 Development Kits

Getting started with designing a beacon first involves the decision of which development kit to use. There are several available from Texas Instruments, which are presented in Figure 9. These kits stretch from small coin cell driven solutions (CC254xDK-MINI, CC2541DK-SENSOR) to fully advanced platforms ideal for prototyping (CC2540DK). More information about these development kits can be found at ti.com/ble.

Figure 9. Development Kits, *Bluetooth* Smart

The development kits can be purchased online at TI store [5].

3.2 Creating a Beacon Application With TI BLE-Stack

The BLE-stack available from Texas Instrument for the CC254x Wireless MCUs provides an easy and reliable implementation of connectable and non-connectable beacons. There are sample applications that can be used as templates when designing a beacon, which are described in Table 5. It is assumed that you are familiar with the IAR Embedded Workbench and the BLE-stack.

Table 5. Beacon Software Examples for CC254x

Example Project	GAP Role	Type	Device support
SimpleBLEPeripheral	Peripheral	Connectable	CC2540, CC2540T, CC2541, CC2541-Q1
SimpleBLEBroadcaster	Broadcaster	Non-connectable	CC2540, CC2540T, CC2541, CC2541-Q1

There is also a generic broadcaster sample application specifically designed for CC2543 that operates in non-connectable broadcasting mode [6]. For the CC2543, there is a reference design for a complete broadcaster solution [7].

SimpleBLEPeripheral is thoroughly described in the software Development Guide [8] and is generally the best starting point if implementing a connectable beacon. The SimpleBLEBroadcaster is a simplified version of SimpleBLEPeripheral, which only supports non-connectable beacons. The API to enable beacon functionality is the same for these projects so the following code examples apply for both, although SimpleBLEBroadcaster (BLEv1.4) is used as the referred example project. There are two hardware platforms defined for the SimpleBLEBroadcaster: CC2541 and CC2541DK-MINI Keyfob. For the purpose of this application report, CC2541EM is assumed to be used. This is configured by the workspace dropdown list as shown in Figure 10. A CC2540 version of the project also exists with similar build options.

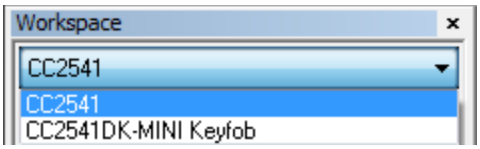

Figure 10. Build Options for SimpleBLEBroadcaster

The application is implemented in SimpleBLEBroadcaster.c where the broadcasting data is defined as advertData.

```
static uint8 advertData[] =
{
  // Flags; this sets the device to use limited discoverable
  // mode (advertises for 30 seconds at a time) instead of general
  // discoverable mode (advertises indefinitely)
  0x02,                                 // length of this data
  GAP_ADTYPE_FLAGS,
  GAP_ADTYPE_FLAGS_BREDR_NOT_SUPPORTED,

  // three-byte broadcast of the data "1 2 3"
  0x04,                                 // length of this data
  GAP_ADTYPE_MANUFACTURER_SPECIFIC,        // manufacturer spec. adv. data type
  1,
  2,
  3
};
```

The default advertised data includes the mandatory Flags followed by 3 bytes of Manufacture-Specific Data (numbers 1, 2 and 3). The Manufacture-Specific Data can be modified to be something else, although note to update the length of the data as well, if necessary. Depending of the hardware platform, advertise on start-up is setup differently. For the CC2541 Build, which is CC2541 generic, default variable for this is set to TRUE. In the sample application, the advertise type is default set to constant define GAP_ADTYPE_ADV_SCAN_IND. This allows Scan Request/Response during advertise event. To disallow that along with reducing the power consumption, advType can be changed to constant define GAP_ADTYPE_ADV_NONCONN_IND instead. Doing that change also allows you to omit the 3 Bytes of Flags as well.

```
// For other hardware platforms, device starts advertising upon initialization
uint8 initial_advertising_enable = TRUE;

// Use non-connectable advertisements
uint8 advType = GAP_ADTYPE_ADV_SCAN_IND;
```

The configured variables are then committed to the GAP layer, which sets up the BLE stack. Note that the *advertEnable* is not instantaneously triggered, especially not during the initiation of the application (simpleBLEBroadcaster_Init). The advertise will start when the protocol stack runs, at a later instance.

```
GAPRole_SetParameter( GAPROLE_ADVERT_ENABLED,sizeof(uint8), &advertEnable );
GAPRole_SetParameter( GAPROLE_ADVERT_DATA, sizeof(advertData), advertData );
GAPRole_SetParameter( GAPROLE_ADV_EVENT_TYPE, sizeof(uint8), &advType );
```

The advertising interval is default set to 100 ms although it can be increased up to 10.24 seconds, which is the maximum allowed by the core specification. If longer intervals are needed, it is possible to manually enable and disable the advertisement with the use of an OSAL timer, as an example.

```
// Advertising interval (units of 625us, 160=100ms)
#define DEFAULT_ADVERTISING_INTERVAL          160
```

To ensure discovery of broadcasts, there is a general rule that the advertisement Interval + 10 should be less than scan window of the observer device. This means that the beacon must be designed with the peer device capabilities in mind. Otherwise, it is possible that it will take a very long time to receive broadcasted packets. This implies that a lower broadcast interval will allow broadcasted data to be observed more quickly, although it requires more power due to the more frequent wake ups. The interval is setup with following API:

```
uint16 advInt = DEFAULT_ADVERTISING_INTERVAL;

GAP_SetParamValue( TGAP_LIM_DISC_ADV_INT_MIN, advInt );
GAP_SetParamValue( TGAP_LIM_DISC_ADV_INT_MAX, advInt );
GAP_SetParamValue( TGAP_GEN_DISC_ADV_INT_MIN, advInt );
GAP_SetParamValue( TGAP_GEN_DISC_ADV_INT_MAX, advInt );
```

For more information regarding application architecture and API description, see the software Development Guide [8].

By using the TI Packet Sniffer [9], the broadcasted data can be verified (as shown in Figure 11). The illustration shows a packet on channel 37 (0x25) that is connectable (ADV_IND). The AdvA value is the IEEE address and advData includes Flags (0x01) and Manufacturer-Specific Data (0xFF).

Channel	Access Address	Adv PDU Type	Adv PDU Header				AdvA	AdvData	CRC	RSSI (dBm)	FCS
			Type	TxAdd	RxAdd	PDU-Length		02 01 04 04			
0x25	0x8E89BED6	ADV_DISCOVER_IND	6	0	0	14	0x90D7EBB1E02B	FF 01 02 03	0xA7C13C	-76	OK

Figure 11. Packet Sniffer v2.18.1, SimpleBLEBroadcaster Packet Over the Air

Note that in the example code, based on BLEv1.4, the Manufacturer-Specific Data does not include the company code, which is the incorrect format according to the core specification. To correct this, simply add a company code to the AdvertData within the application software as shown below. Also remember to update the length of the Manufacturer-Specific Data, which also includes the flag for the data type as well.

```
// GAP - Advertisement data (max size = 31 bytes, though this is
// best kept short to conserve power while advertising)
static uint8 advertData[] =
{
  // Flags; this sets the device to use limited discoverable
  // mode (advertises for 30 seconds at a time) instead of general
  // discoverable mode (advertises indefinitely)
  0x02,                                 // length of this data
  GAP_ADTYPE_FLAGS,
  GAP_ADTYPE_FLAGS_BREDR_NOT_SUPPORTED,

  // three-byte broadcast of the data "1 2 3"
  0x06,                                // length of this data
  GAP_ADTYPE_MANUFACTURER_SPECIFIC,       // manufacturer spec. adv. data type
  0x0D,                                // TI Company code
  0x00,                                // TI Company code
  1,
  2,
  3
};
```

4 References

1. Bluetooth.org, Core specification v4.1: https://www.bluetooth.org/en-us/specification/adopted-specifications
2. Bluetooth.org, Supplements to the Core Specification (CSS) v.4: https://www.bluetooth.org/en-us/specification/adopted-specifications
3. Bluetooth.org, Assigned Numbers, GAP, 22.10.2014 13:37: https://www.bluetooth.org/en-us/specification/adopted-specifications
4. Indoor Light Energy Harvesting Reference Design for Bluetooth Low Energy (BLE) Beacon Subsystem: http://www.ti.com/tool/tida-00100
5. TI Online Store: https://estore.ti.com/
6. CC2543 miniBLEBroadcaster: http://www.ti.com/product/cc2543/toolssoftware
7. Mini Bluetooth® Low Energy Broadcaster: http://www.ti.com/tool/tidc-mini-bluetooth-low-energy-broadcaster
8. TI BLE Software Developer's Guide, included with the BLE SDK Installer: www.ti.com/ble-stack
9. TI Packet Sniffer: http://www.ti.com/tool/packet-sniffer

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information 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.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265