The Many Communication Modes of Bluetooth® LE

$ADVB_L$ – Legacy Advertising

Understanding Legacy Advertising ($ADVB_L$)

When Bluetooth Low Energy (LE) first appeared in version 4.0 of the Bluetooth Core Specification, it had only two different communication modes. The first was the connection-oriented LE-ACL mode, which was covered in another of the articles in this series. The second was informally called *advertising*.

This original form of advertising is defined technically by one of the Bluetooth LE logical transports, Advertising Broadcast or ADVB for short.

In a later version of the Bluetooth Core Specification (v5.0), ADVB was enhanced to add a more sophisticated set of capabilities that are collectively known as Extended Advertising. At this point, the original advertising capability became known as Legacy Advertising to allow a distinction to be made between these two variants.

The legacy variant and the newer extended version of the ADVB logical transport are both big enough subjects to warrant covering each separately, and so this article covers legacy advertising only and uses $ADVB_L$ as a shorthand label for Advertising Broadcast (legacy variant).

Overview

Communication using $ADVB_L$ works very differently to communication using LE-ACL connections. The most obvious difference is that this logical transport provides a form of *connectionless* communication. This means that in contrast to LE-ACL connection-oriented communication, there is no prior agreement made between devices, and they each transmit and receive independently of each other.

Advertising involves one device transmitting packets at intervals while one or more other devices receive those packets by *scanning*. The $ADVB_L$ communication mode is more involved than this however, as we shall learn in the remainder of this article.

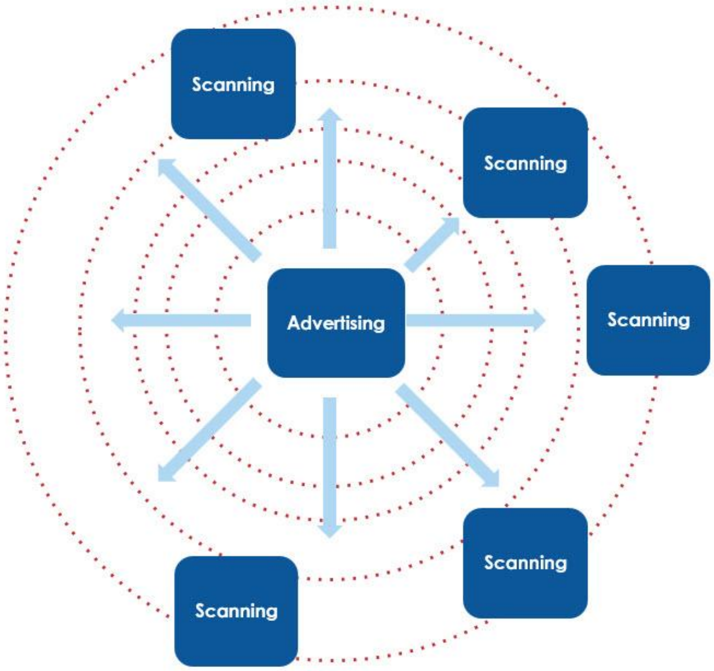

Figure 1 Advertising and Scanning

$ADVB_L$ was originally designed with two use cases in mind. *Device discovery* was the first of these use cases. *Beaconing*, where a single device broadcasts unchanging information repeatedly was the second use case. Using $ADVB_L$ as a transport for the general, connectionless communication of arbitrary and variable data was always possible but not something it was commonly used for.

Legacy Advertising Variants

When a device advertises, it does more than just transfer data to other devices. It also indicates actions that devices that receive the transmitted packets might be able to take.

The Bluetooth Core Specification defines a number of different types of advertising event. The various event types involve the transmission of different types of protocol data units (PDUs) and may permit receiving devices to respond with PDUs of their own.

There are a few variables that make one advertising event type different to another. These variables are listed and explained in Table 1.

Variable	Explanation
Connectable vs Non-connectable	An advertising device that wishes to indicate that it can accept connection requests from other devices (and so form an LE-ACL connection) performs *connectable advertising* whereas one which does not allow connections performs *non-connectable advertising*.
Scannable vs Non-scannable	An advertising device can indicate that more information can be requested from it by other devices replying to its transmitted advertising packets with a *scan request*. It does this by performing *scannable advertising*. A replying device requesting further information is said to be performing *active scanning*. An advertising device that does not offer further information performs *non-scannable advertising* and receiving devices are said to perform *passive scanning*.
Directed vs Undirected	As suggested by the name of the logical transport, advertising *broadcast*, an advertising device can transmit packets that are intended to be received by any and all devices that are in range and scanning. This is known as *undirected advertising*. But $ADVB_L$ can also be used to transmit packets from the advertising device to one specific device only and this is called *directed advertising*.

Table 1 Advertising event type variables

$ADVB_L$ allows four different combinations of these variables, forming four different advertising event types in total. The list of $ADVB_L$ advertising event types is as follows:

1. connectable and scannable undirected
2. connectable directed
3. non-connectable and non-scannable undirected
4. scannable undirected

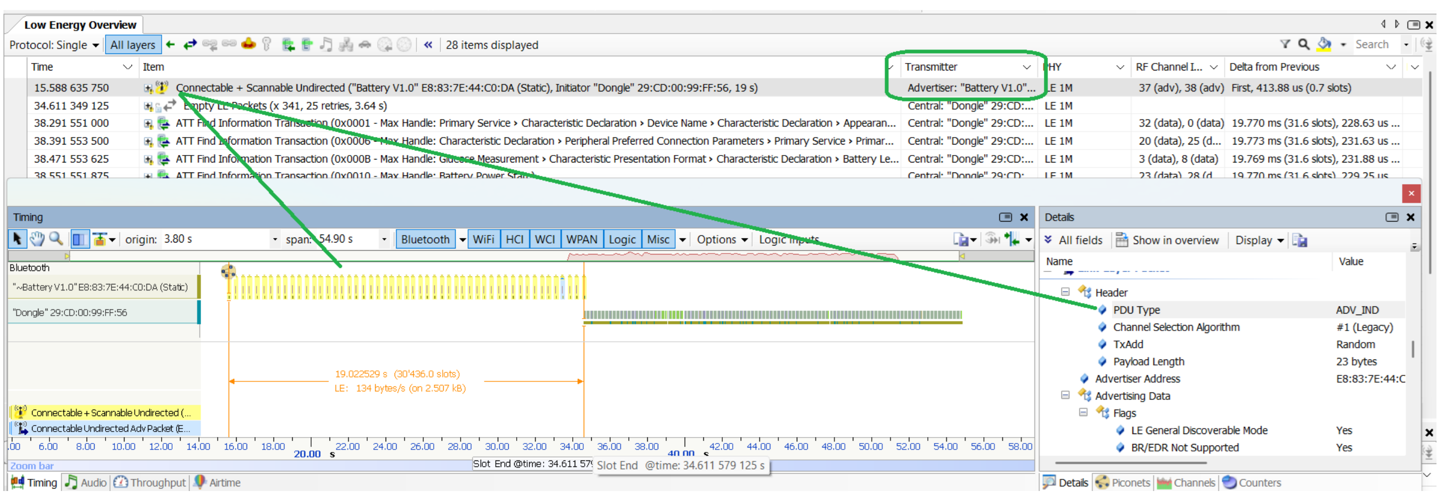

Figure 2 Ellisys Bluetooth analysis software showing a hierarchical view of "Battery" device advertising as connectable and scannable undirected

There's a further property that an advertising device can be said to possess and this is whether or not it is *discoverable*. This is a concept that is defined by the Generic Access Profile (GAP) part of the Bluetooth Core Specification. GAP includes the rules for using Link Layer capabilities for purposes such as the discovery of other devices or for a device to be discovered.

Discoverable is a concept that is sometimes misunderstood.

Any device which is scanning on the right channel can receive any packet that was transmitted by another in-range device. In that sense, all devices could be said to be discoverable. But in the context of the Bluetooth Core Specification and GAP, it has a more nuanced definition and purpose. When a device is in one of the discoverable modes (there are two), it simply means that it *intends* to be discovered. This allows receiving devices to filter devices that do not intend to be discovered from further processing such as presentation to the user within a graphical user interface (GUI). It's important to appreciate that a device that is not discoverable is not *invisible*. This is not a security feature, in other words.

Device Roles

GAP defines four device roles that relate to advertising and scanning:

- **Broadcaster** - In this role, a device transmits but does not receive packets. As such, a Broadcaster cannot be connected to.
- **Observer** - An Observer is the counterpart to the Broadcaster. It receives advertising packets, may reply to request more information but cannot initiate a connection with the Broadcaster.
- **Peripheral** - A GAP Peripheral device transmits advertising packets and can receive packets such as connection requests or scan requests, depending on the advertising event type in use.
- **Central** - A GAP Central is the counterpart to the GAP Peripheral. It receives advertising packets by scanning and may reply with transmitted connection or scan requests depending on the advertising event type in use.

Note that the terms Peripheral and Central are also used by the Link Layer. The meaning of these terms across the two layers is not identical and at the Link Layer only applies when devices are connected. But a GAP Peripheral will assume the Link Layer Peripheral role when connected to, and a GAP Central will assume the Link Layer Central role after establishing a connection to a GAP Peripheral.

Packets and PDUs

The Link Layer defines a general advertising packet format. A series of different PDUs can be sent within the payload of an advertising packet and there is a different PDU type for each advertising event type, as well as request and response PDUs for active scanning and a PDU that is sent to initiate the establishment of a connection.

The type of PDU that can be sent depends on the advertising event type and device role as summarized in Table 2.

PDU	When Transmitted	Transmitted By
ADV_IND	Connectable and scannable undirected event.	Broadcaster or Peripheral
ADV_DIRECT_IND	Connectable directed event.	Broadcaster or Peripheral
ADV_NONCONN_IND	Non-connectable and non-scannable undirected event.	Broadcaster or Peripheral
ADV_SCAN_IND	Scannable undirected event.	Broadcaster or Peripheral
SCAN_REQ	In reply to ADV_IND or ADV_SCAN_IND.	Observer or Central
SCAN_RSP	In reply to SCAN_REQ.	Broadcaster or Peripheral
CONNECT_IND	In reply to ADV_IND or ADV_DIRECT_IND.	Central

Table 2 Advertising PDUs

Scheduling

In common with other Bluetooth communication modes, $ADVB_L$ includes rules about when the radio can be used. These rules apply to both devices that advertise, being in either the Broadcaster or Peripheral role, and devices that can transmit scan requests (Observer and Central) or connection requests (Central only).

Advertising

Legacy advertising radio activity takes place within *advertising events*. An advertising event starts when a device first enters the Link Layer advertising state. The time when the next advertising event starts is calculated from the duration of a timing parameter called the *advertising interval* plus a randomly generated time value called *advDelay*. The advertising interval has a static value in the inclusive range of 20 ms to 10.24 seconds and is set in response to a configuration request that is made by the application. The advDelay value is randomly selected afresh at the start of every advertising event and is in the range 0 to 10 ms inclusive.

When an *advertising event* starts, the Broadcaster or Peripheral device transmits one, two or three copies of the same packet on different radio channels one at a time and with a gap of no more than 10 ms between the start of successive packets. Legacy advertising packets are short and take less than 400 µs to transmit so 10 ms is plenty of time to get this done and perhaps more besides as we shall see.

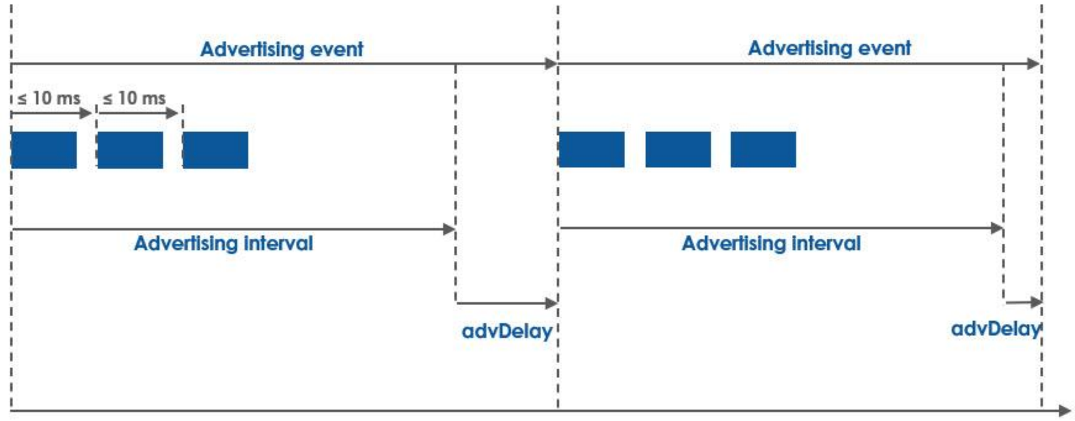

Figure 3 Timing of advertising activity in terms of intervals, delays and events

Note: There's an exception to this rule. See the *Did You Know?* section below for an interesting point about high vs. low duty cycle advertising.

The inclusion of the randomly generated advDelay variable in the scheduling of advertising events results in advertising activity being perturbed in time. In other words, legacy advertising follows an irregular timing schedule in contrast to the regular timing pattern that governs events in an LE-ACL connection. The reason behind this concerns the possibility of collisions.

A collision occurs when the transmission of a packet by two or more different devices is made on the same radio channel in an overlapping timeframe. The effect of a collision is to corrupt the packets. So, by randomizing advertising schedules, the probability that persistent, recurring collisions between different devices will occur is much reduced.

Scanning

Scanning on a single channel takes place periodically in accordance with a timing parameter called the *scan interval*. How long a device scans the selected channel for is controlled by a parameter called the *scan window*.

When using active scanning, an Observer or Central device may request more information by sending a SCAN_REQ PDU in reply to an ADV_IND or ADV_SCAN_IND PDU on the same radio channel. On receiving the SCAN_REQ PDU, the advertising device will send additional data in a SCAN_RSP PDU, again on the same radio channel.

A gap equal to the Inter-Frame Space (IFS) must be left before transmitting SCAN_REQ and SCAN_RSP PDUs and the total elapsed time between the start of successive ADV_IND or ADV_SCAN_IND PDUs must still not exceed 10 ms. This is depicted in Figure 4.

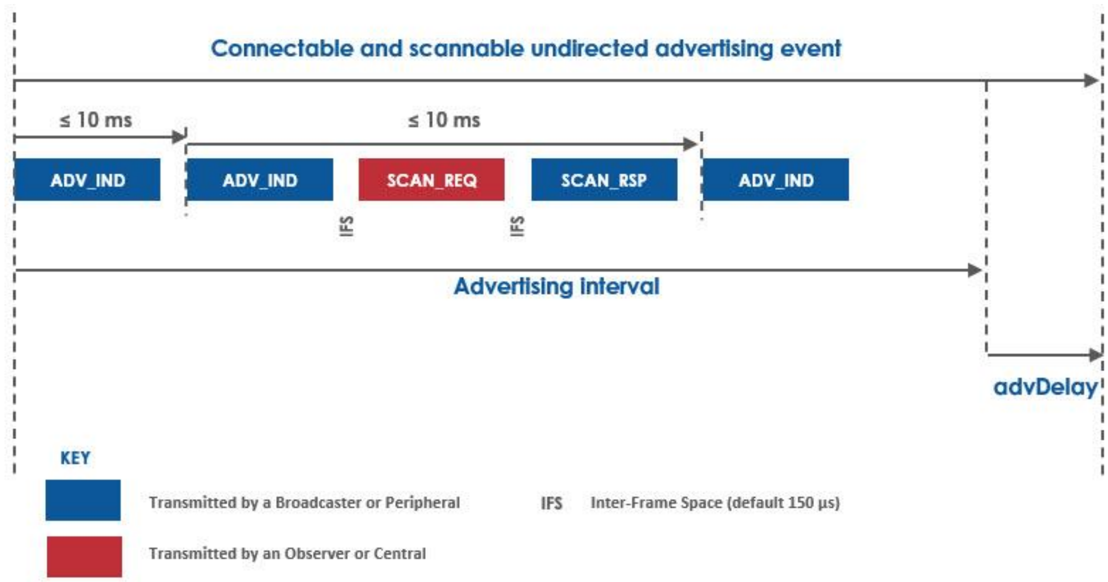

Figure 4 Scheduling of SCAN_REQ and SCAN_RSP PDUs

Initiating

A device that requests a connection is in the Link Layer Initiating state.

A device scanning for advertising packets in a connectable advertising event can reply to an ADV_IND or ADV_DIRECT_IND PDU with a CONNECT_IND PDU. This represents the scanning device (in the Central role) asking to form a connection with the advertising device (in the Peripheral role).

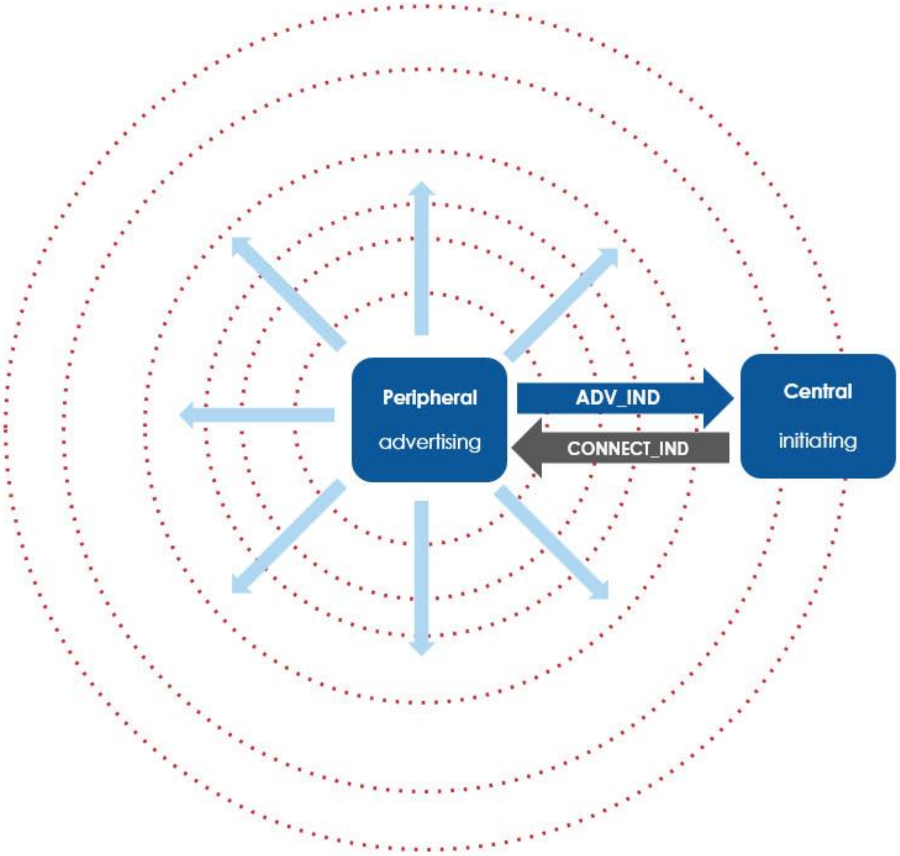

Figure 5 Central device responding to an ADV_IND PDU with a CONNECT_IND PDU

Addressing

All legacy advertising PDUs include a field that contains the Bluetooth device address of the advertising device. The name of the field is AdvA.

ADV_IND, ADV_NONCONN_IND and ADV_SCAN_IND PDUs, sent by an advertising device in undirected advertising events, do not contain a target Bluetooth device address. These PDUs by definition are not intended for any one device. They do however contain a field called AdvData and it is this field that application data can be placed in.

PAYLOAD	
AdvA (6 octets)	AdvData (0-31 octets)

Figure 6 Fields in ADV_IND, ADV_NONCONN_IND and ADV_SCAN_IND PDUs

ADV_DIRECT_IND PDUs are sent in connectable and directed advertising events and are intended for a specific device. Therefore, this PDU includes a field called TargetA which contains the Bluetooth device address of the target device. Note that the AdvData field is absent from this PDU type.

PAYLOAD	
AdvA (6 octets)	TargetA (6 octets)

Figure 7 Fields in ADV_DIRECT_IND PDUs

SCAN_REQ and CONNECT_IND PDUs contain both the address of the source, scanning device and the remote address of the advertising device.

SCAN_RSP PDUs contain the address of the advertising device which is responding to the received SCAN_REQ PDU but do not include an address for the target device.

Radio Channels

ADVB$_L$ divides the 2.4 GHz band into 40 channels, each 2 MHz wide and uses 3 specific channels for advertising. These channels are known as the primary advertising channels.

The location of the primary advertising channels was carefully chosen to avoid Wi-Fi transmissions and to be a sufficient distance from each other to maximize the likelihood that at least one of these important channels will be able to function well.

The channel index numbers of the primary advertising channels are 37, 38 and 39 respectively.

37	2402 MHz
0	2404 MHz
1	2406 MHz
2	2408 MHz
3	2410 MHz
4	2412 MHz
5	2414 MHz
6	2416 MHz
7	2418 MHz
8	2420 MHz
9	2422 MHz
10	2424 MHz
38	2426 MHz
11	2428 MHz
12	2430 MHz
13	2432 MHz
14	2434 MHz
15	2436 MHz
16	2438 MHz
17	2440 MHz
18	2442 MHz
19	2444 MHz
20	2446 MHz
21	2448 MHz
22	2450 MHz
23	2452 MHz
24	2454 MHz
25	2456 MHz
26	2458 MHz
27	2460 MHz
28	2462 MHz
29	2464 MHz
30	2466 MHz
31	2468 MHz
32	2470 MHz
33	2472 MHz
34	2474 MHz
35	2476 MHz
36	2478 MHz
39	2480 MHz

Figure 8 Primary advertising channels (shown in light blue)

Applications can opt to have one, two or all three of the primary advertising channels used and this is indicated in a data structure called the advertising channel map.

In each advertising event, the link layer transmits a copy of the same advertising packet on each of the channels that are in use according to the advertising channel map. The ordering of channels is randomly selected at each advertising event.

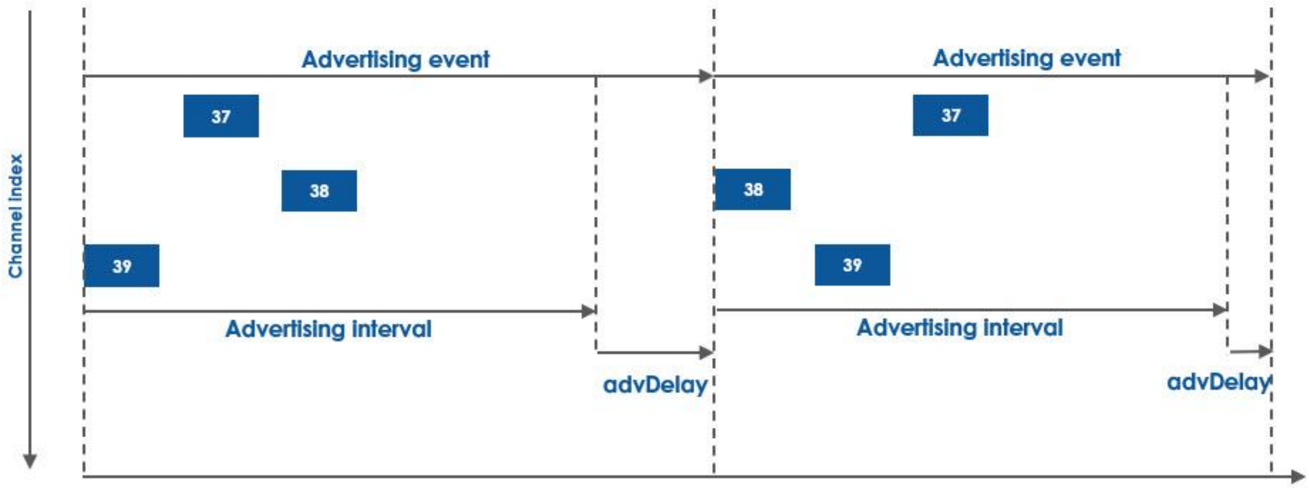

Figure 9 Advertising on randomly selected primary advertising channels

The Physical Layer

The legacy advertising PDUs, namely ADV_IND, ADV_DIRECT_IND, ADV_NONCONN_IND, ADV_SCAN_IND, SCAN_REQ, SCAN_RSP, and CONNECT_IND can only use the LE 1M PHY and therefore are transmitted with a symbol rate of one million symbols per second.

Application Concerns

Applications have their own sets of requirements and priorities. In this section we'll consider a selection of those application issues that have a bearing on using Bluetooth $ADVB_L$ connectionless communication.

Configuration

Applications can specify or suggest a number of parameters that affect how legacy advertising functions. By using a Host Controller Interface (HCI) command, *LE Set Advertising Parameters*, applications can for example:

- Specify a minimum and maximum advertising interval.
- Specify the type of advertising to be performed.
- Specify the address of a peer (remote) device.
- Indicate which of the three primary advertising channels are to be used.

The advertising interval is selected by the scheduling algorithm of the product, an implementation concern rather than one which is specified in full by the Bluetooth Core Specification. If it is not possible to arrive at a value that falls within the application's requested range, an error is returned by the Controller to the application in an HCI event.

The advertising type configuration parameter determines which advertising event type and associated PDU is to be used by the Link Layer.

A peer address is only a concern if directed advertising is to be performed.

The advertising channel map uses a series of 3 bits to indicate which of the three primary advertising channels are to be used for packet transmission.

It should be noted that while the HCI defines a command and parameters to be used to configure legacy advertising, application developers may not have access to all parameter values via their platform's APIs. API documentation should always be consulted to find out what the capabilities and constraints are for applications being developed for a particular platform.

Application Data

Three of the four legacy advertising PDUs include a field called AdvData. It is in this field that applications can place data that they wish to communicate to other devices.

As shown previously in Figure 6, a maximum of 31 bytes can be stored in the variable length AdvData field but as will be explained, not all of it is actually application data. This is because applications can place a series of data items in the AdvData field and each of them must be encoded within a data structure that is known as an AD Type.

All AD Types have the same three field structure:

- **Length** - a single byte whose value indicates the number of bytes occupied by the other two fields.
- **Type** - a single byte whose value acts as an identifier for the type of data in the final of the three fields.
- **Value** - a variable number of bytes (the value of Length field minus 1) that constitute the application data itself.

For example, Figure 10 shows an advertising packet transmitted in a connectable and scannable undirected event. The payload is presented in hexadecimal format and is 11 bytes long in total. This is the AdvData field.

Figure 10 Advertising packet in the Ellisys Bluetooth Analyzer

Note: *Heart Rate Monitor advertising packets generated using a Nordic Semiconductor nRF54L15DK development kit.*

With reference to the Length/Type/Value structure of AD Type sequences, we can break the payload down as follows:

Length	Type	Value (hex)
02	01	06
07	03	0D18 0F18 0A18

Table 3 Raw AD Types data in the advertising PDU AdvData field

We could now look up the types and their structure in the relevant specifications but using an Ellisys Bluetooth Analyzer, we don't have to because this is automatically done for us. Figure 11 shows that the advertising packet contains an ADV_IND PDU, and decodes the fields contained within AdvData, giving us their values and their meaning.

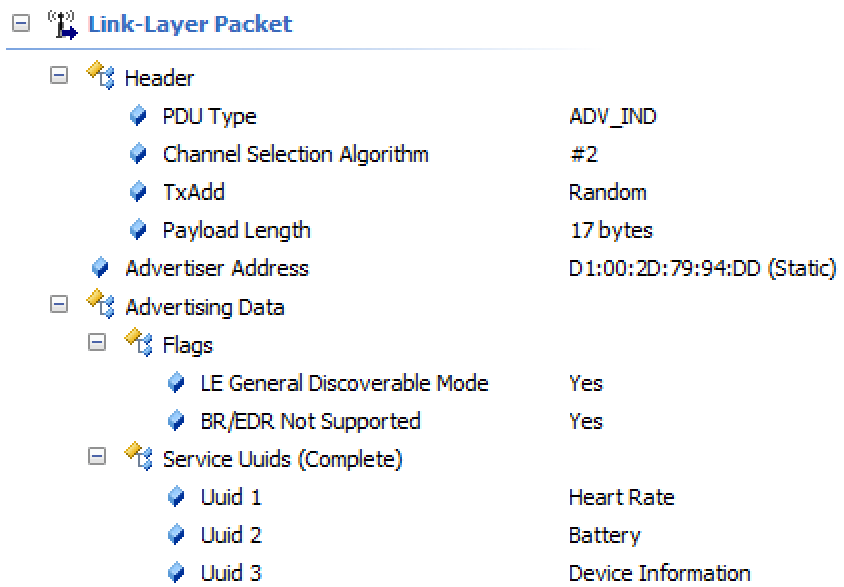

Figure 11 Decoded content of the AdvData field

The Flags field indicates that this device is in one of the two discoverable modes.

The Bluetooth LE Generic Attribute Profile (GATT) uses a system of Universally Unique Identifiers (UUIDs) to identify the capabilities (**services**) and data (*characteristics* and *descriptors*) implemented within a device. The Service UUID values in this advertising packet indicate that the advertising device supports GATT services that provide heart rate information, battery level information, and general device information. For any scanning device seeking to discover other devices that are relevant to its application, this information is useful. And with a little knowledge, we can tell at a glance from the information the Ellisys protocol analyzer has provided that the discovered device supports the Bluetooth Heart Rate Profile.

Note that AD types are defined in a specification called the Core Specification Supplement and type identifiers in the Bluetooth SIG Assigned Numbers document.

General Data Communication

$ADVB_L$ can be used as a general transport for the connectionless communication of data. The AdvData field and its use of AD Types as the packaging structure for all data items means that the amount of data that can be transmitted in an advertising PDU is quite small, but it can still be usefully harnessed in many use cases.

The main reasons that $ADVB_L$ might be selected for use as a general communications transport include:

- **Simplicity** - no connection needs to be established beforehand and therefore no device discovery needs to take place.
- **Scalability** - A device can accommodate a limited number of concurrent connections and so communicate with only a small number of devices at the same time using connections. In contrast, $ADVB_L$ allows each transmitted packet to be received by any number of devices.

Consider the following example. A new electricity monitor product was to be based on a constrained microcontroller, capable of supporting only one connection at a time. But the requirements for this product include that it be able to communicate real-time power consumption data to multiple devices at the same time, including multiple smartphones and a data logging server. So, $ADVB_L$ was chosen as the transport and the microcontroller broadcasts power values which are received simultaneously by anyone using the associated smartphone app and by the Linux data logging server in which all readings are stored so that historical analysis of power consumption is possible.

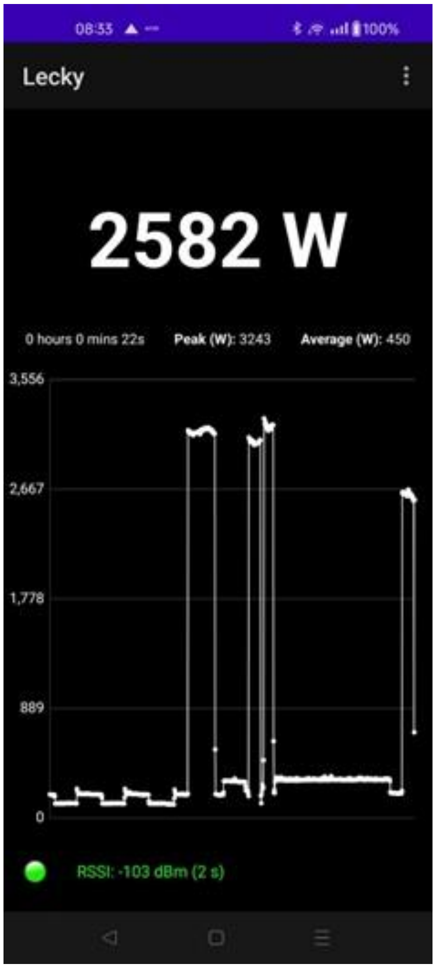

Figure 12 $ADVB_L$ broadcast data being received and used in a smartphone app

The Ellisys Protocol Analyzer software allows us to examine the broadcasts from this device and an example packet, captured during testing, as shown below in Figure 13.

Link-Layer Information	
Sniffer Radio	
RSSI	-55.0 dBm
RX Quality	Average
RF Gain	6.0 dB
RF Channel	
RF Channel Number	12
RF Channel Index	38 (adv)
Initial Center Frequency Offset	+31.25 kHz
Link Layer	
PHY	LE 1M
Coding Scheme	Uncoded (1 Mbps)
Access Address	0x8E89BED6
CRC Initial Seed	0x555555
Physical Channel	Advertisement ("LECKY_TEST" CE:B3:1E:74:E5:F9 (Static))
Timing	
Start Time	23.407 523 625
Duration	296 us
Delta from Previous	582.88 us (0.9 slots)
Devices	
Originator	Advertiser
Transmitter	Advertiser: "LECKY_TEST" CE:B3:1E:74:E5:F9 (Static)
Receiver	Initiator: "Scanning Device"
Advertiser Address	CE:B3:1E:74:E5:F9 (Static)
Initiator Address	Unknown BD_ADDR
Link-Layer Packet	
Header	
PDU Type	ADV_IND
Channel Selection Algorithm	#2
TxAdd	Random
Payload Length	27 bytes
Advertiser Address	CE:B3:1E:74:E5:F9 (Static)
Advertising Data	
Flags	
LE General Discoverable Mode	Yes
BR/EDR Not Supported	Yes
Local Name	
Complete Local Name	"LECKY_TEST"
Manufacturer Specific Data	
Company ID	Reserved
Manufacturer Specific Data	03 8D

Figure 13 ADV_IND packet from the electricity monitor Peripheral device

Note: Electricity monitor advertising packets were generated with a BBC microbit V2 which contains a Nordic Semiconductor nRF52833.

There's an abundance of useful information displayed in Figure 13:

- We can see that the packet was transmitted on general advertising channel 38.
- The advertising device's address is CE:B3:1E:74:E5:F9.
- The advertising event type is connectable and scannable undirected and the PDU type ADV_IND.
- The device is discoverable.
- The AdvData field contains the Complete Local Name AD Type with the value "LECKY_TEST" and the Manufacturer Specific Data AD Type with two sub-fields.

The Manufacturer Specific Data AD Type is worth special mention. Any arbitrary data for which there is no standard AD Type defined may be placed in this field. The rule for its use is that the manufacturer's company ID (issued by the Bluetooth SIG) be included before the application data itself in the Manufacturer Specific Data sub-field so that devices that receive it can decode it with respect to that company's specification. In the example shown in Figure 13, a power value in Watts has been placed in the field and decoding the hex number 0x038D as a big-endian value reveals that the electricity monitor is reporting power use of 909 Watts.

But perhaps the best example of $ADVB_L$ being used as a general transport for data is Bluetooth Mesh networking. Bluetooth Mesh uses $ADVB_L$ as the transport for mesh messages that travel across a network, allowing sensor data to be harnessed, lights to be controlled and so on. This makes for compelling proof that $ADVB_L$ has what it takes to support general connectionless communication. Note that in the Bluetooth Mesh Protocol Specification, the use of $ADVB_L$ is defined as a *bearer*.

Energy Efficiency

For the advertising device, the transmission of small packets has a low energy cost and the only relevant variable to consider therefore is the number of packets transmitted. This can be controlled by the advertising interval value and the number of channels used. Note however that a long advertising interval will have consequences for the responsiveness of scanning applications and using less than the three available advertising channels may make communication less reliable. But if energy efficiency is the priority, then these are the variables to consider.

For a scanning device, energy efficiency is more problematic. The connectionless nature of $ADVB_L$ communication means that the scanner does not know when to expect advertising to take place and the randomized advertising schedule doesn't exactly help. Therefore, how often scanning takes place (the scan interval) and for how long (the scan window) are parameters which need to be carefully considered so that data is received sufficiently reliably without the need to scan for excessively long periods with the energy cost that this would incur.

Reliability

Advertising packets contain a Cyclic Redundancy Check (CRC) field, and this allows corruption of data in-flight to be detected.

In other respects, reliable connectionless communication using $ADVB_L$ requires thought and for the application layer to take certain steps if reliability is an important requirement. It might seem odd to suggest that there might be times when reliability is not important, but this is the case. It's unlikely to be essential that every transmission of the same URL from a beacon in a shopping mall is received by a smartphone application, for example.

The issue that must be considered is the fact that the advertising schedule of the Broadcaster or Peripheral is not coordinated in any way with the scanning schedules of the Observer or Central devices. Furthermore, a packet can only be transmitted on one channel at a time only and scanning can take place on one channel at a time. For a packet to be received, scanning must take place on the right one of three channels at the right time and for a sufficient duration. This can make $ADVB_L$ a bit hit and miss as a data transport. Figure 14 below provides an example.

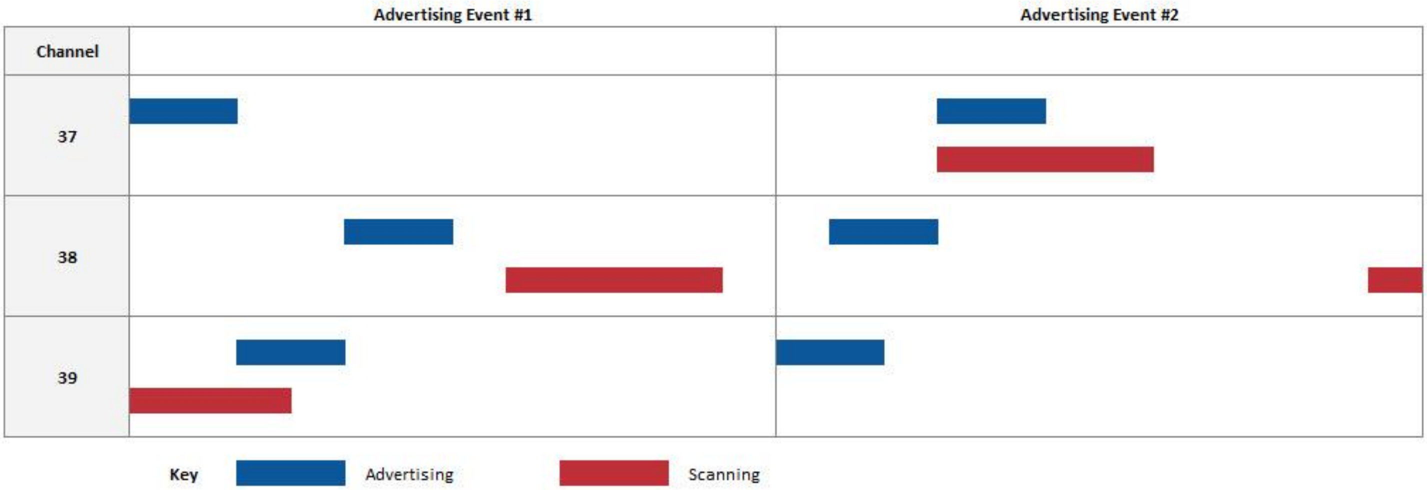

Figure 14 Uncoordinated scanning and advertising schedules

Here we can see that during the first advertising event, scanning first takes place on channel 39 but only overlaps for the first part of the transmitted advertising packet. Scanning takes place on channel 38 when no packet is being transmitted. In the second advertising event, scanning on channel 37 fully overlaps the transmission of a packet and so it is received this time.

Bluetooth Mesh uses a special technique to improve the reliability of $ADVB_L$ as a bearer for mesh messages. Typically, messages are transmitted in advertising packets multiple times in rapid bursts. The repeated transmission of each message increases the odds that it will be received dramatically and is an effective technique to use for connectionless communication in general.

Security

Confidentiality of Data

Until the release of Bluetooth Core Specification v5.4 there was no standard mechanism for safeguarding the confidentiality of application data transmitted in advertising packets and such security requirements had to be dealt with by the application.

Bluetooth Core Specification v5.4 introduced a new feature called Encrypted Advertising Data which provides a standardized mechanism for an advertising device to encrypt data for inclusion in the AdvData and importantly, to share encryption key parameters with trusted devices.

For an advertising device to be able to use the Encrypted Advertising Data feature it must be able to adopt the Peripheral role and therefore be able to accept connections. This is because the sharing of encryption key parameters takes place over a LE-ACL connection that is itself encrypted, and as such, devices must also have been paired.

The advertising device encrypts data fields it wishes to protect from eavesdropping as a single composite sequence of AD Types and places the resultant cipher text in an AD Type called Encrypted Data. Only those scanning devices that are trusted and have obtained the encryption key parameters can decrypt this part of the AdvData field.

Privacy

Link Layer PDUs used for advertising, scanning and other purposes often include a field in which the address of the transmitting device might be placed (e.g. AdvA in advertising PDUs). Sometimes the inclusion of the device's address is mandatory and sometimes it is optional. A device address is an identifier and to work as such, it needs to be static.

Devices which repeatedly transmit static, unchanging data are vulnerable to being tracked. In security circles this is classified as a *privacy issue*.

To mitigate this issue as it applies to Bluetooth device addresses, a special type of address called a *private address* is defined in the specification. A private address automatically changes at intervals. How often depends on the implementation but the core specification recommends an interval of about 15 minutes.

If it's necessary to both protect the privacy of a device and allow it to be reliably identified by *trusted* devices, a special type of private address, the *resolvable private address* (RPA) is defined. If an RPA is used, then a device also has a hidden *identity address* which acts as a reliable and unchanging identifier for the device. When used along with an RPA, the identity address is never revealed but devices can translate or *resolve* an RPA into its identity address if and only if the two devices have been paired.

Note that whenever a private address changes, one of the inputs to the encrypted advertising data encryption procedure (see Confidentiality of Data above), known as a *nonce*, also changes and therefore so does the value of the encrypted data. This too helps protect the privacy of the device.

Communication Mode Properties

The first article in this series on The Many Communication Modes of Bluetooth LE introduced a set of properties that can be useful in comparing one communication mode with another. Here is the table of properties for legacy advertising ($ADVB_L$).

Property	Comment
Topology	Undirected: one-to-many (1:m) or Directed: one-to-one (1:1)
Transmitters vs Receivers	This depends on the advertising event type and device role(s) in use. *See Device Roles section above.*
Application Data Direction	One-way. Application data can be transmitted only from the advertising device to scanning devices.
Connected or Connectionless?	Connectionless.
Data and Time	Asynchronous.
Receiver Concurrency	An unlimited number of scanning devices can receive undirected advertising packets at the same time. Directed advertising means transmitted packets are only relevant to one scanning device.
Radio Channels	Uses up to 3 of the general advertising channels. Channels to be used are determined by the application. Channel ordering is randomized.
Scalability	Undirected advertising can reach a theoretically unlimited number of devices so in this respect, $ADVB_L$ is highly scalable. Application data throughput over $ADVB_L$ is relatively slow due to the small AdvData payload size and limitations in advertising interval values that are permitted.
Choice of PHY	LE 1M only.

Table 4 $ADVB_L$ Properties

Did You Know?

As we approach the end of this article we'll close with a series of interesting and useful additional points about legacy advertising.

Did you know that the Controller implements several filter policies?

Applications can configure Controller filtering criteria to be applied when in the scanning or initiating states. For example, the *scanning filter policy* can be configured so that only the first advertising packet received from each device is passed to the Host. As such, duplicate advertising packets from a device that has already been discovered are discarded. Similarly, the *initiator filter policy* filters connectable advertising packets in various ways. Filter policies make for more efficient processing by the Host since it is not sent irrelevant or already received data by the Controller.

Did you know that the Bluetooth Core Specification v6.0 introduced a new feature called Monitoring Advertisers?

Controller filter policies can improve efficiency but there's a potential downside. If no packet filtering is applied by the Controller, then the Host can deduce when a device of interest is present and when it moves in and out of range. This can be invaluable information but gained at the expense of the need to process duplicate packets.

The Monitoring Advertisers feature allows the Host to configure the Controller so that when a device of interest moves in and out of range, the Host is notified by an HCI event sent by the Controller. This allows the application to ask the Controller to filter duplicate packets and to remain aware of the presence or absence of a device without incurring the overhead of processing every received advertising packet.

Did you know that there are two different Discoverable modes?

The Generic Access Profile (GAP) defines limited discoverable mode and general discoverable mode. A device can be in limited discoverable mode for no longer than 180 seconds. There is no limit to how long a device can be in general discoverable mode.

The discoverable mode that a device is in is indicated by bits of the Flags AD Type in advertising PDUs.

The reason for the two modes is that it allows the prioritization of devices in the limited discoverable mode over those in the general discoverable mode.

Did you know that it may be possible to change the inter-frame space time parameter used in connections?

One of the uses of directed advertising is to prompt a target device to connect quickly to the advertising device. To that end, directed connectable advertising can be performed in either *high duty cycle mode* or *low duty cycle mode*. In high duty cycle mode, the elapsed time between the start of two consecutive ADV_DIRECT_IND PDUs sent on the same advertising channel may not exceed 3.75 ms. In low duty cycle mode, adjacent PDUs transmitted on any channel must not be separated by more than 10 ms. Note that high duty cycle directed advertising is expensive in terms of energy use.

Next in the Series

In this article we've explored the $ADVB_L$ communication mode of Bluetooth LE.

In the next article in this series, we'll take a closer look at the extended advertising ($ADVB_E$) communication mode.