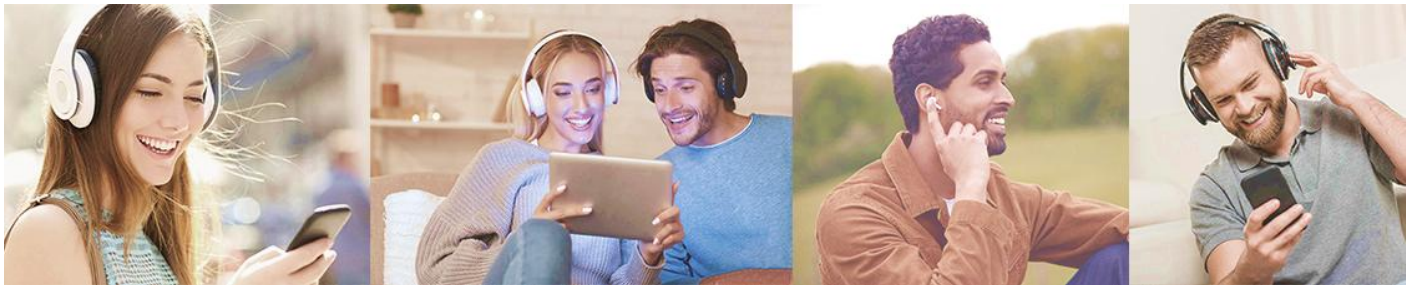
The Many Communication Modes of Bluetooth® LE

ACL Connections



Understanding Bluetooth LE Connections (ACL)

One of the most common Bluetooth communication modes involves the exchange of data over the humble *connection*.

But humble or not, in the Bluetooth Core Specification, a connection is a formally defined technical concept. In this article, we'll examine and learn about some of the key technical aspects of connections in Bluetooth Low Energy (LE) and come to understand what they are, how they work, how they can be made use of by applications, and their pros and cons.

In the Bluetooth Core Specification, communication using connections is defined as a *logical transport* called the Bluetooth LE Asynchronous Connection-oriented Logical Transport, or LE-ACL for short. Its technical details can be found in the Link Layer section of the specification.

Device Roles

When two devices are connected, they each assume one of two roles:

- Central - this is the device that initiated the establishment of the connection. It makes some of the key decisions about how the connection behaves and drives the exchange of packets from the start.
- Peripheral - this is the device that invites others to connect to it. It transmits packets over the connection but only in reply to those that it receives from the Central device.

In Figure 1 below, which shows the Piconet view from the Ellisys Bluetooth software application, several piconets are active and are transferring data. Each includes a central device and a single peripheral – central devices have a blue border. Advertising and other broadcast traffic is filtered from this view.

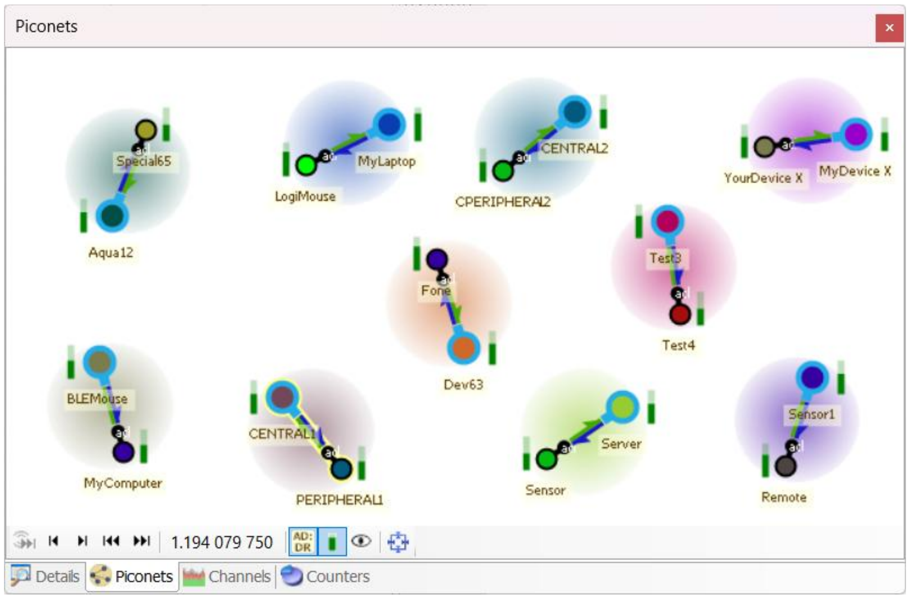


Figure 1 Piconet View from Ellisys Bluetooth Analyzer Application Showing Multiple Connections.

Establishing a Connection

Connections between devices don't exist. Until they do.

Before two devices can form a connection, one must discover the other and then establish a connection to it. Device discovery involves a Peripheral device *advertising*. This involves a different Bluetooth logical transport called *Advertising Broadcast* (ADVB) which will be examined in more detail in later articles in this series. In essence though, the Peripheral broadcasts small packets that contain information about the Peripheral and which act as an invitation to relevant devices to connect.

Note: Strictly speaking, the terms Central and Peripheral only apply to devices that have already connected but we're using the terms here in the description of the discovery and connection establishment process for simplicity.

A Central device wishing to find a suitable Peripheral to connect to, switches its radio into receive mode and scans (listens) for advertising packets. On discovering what seems to be a suitable device or devices, it's common for the Central device to present details to the user, have the user select the device to connect to, and then for the Central device to transmit a packet containing a protocol data unit (PDU) which acts as a request to connect. From this point on, the two devices are said to be in the Link Layer Connection State. Figure 3, Figure 4, and Figure 5 below illustrate this process.

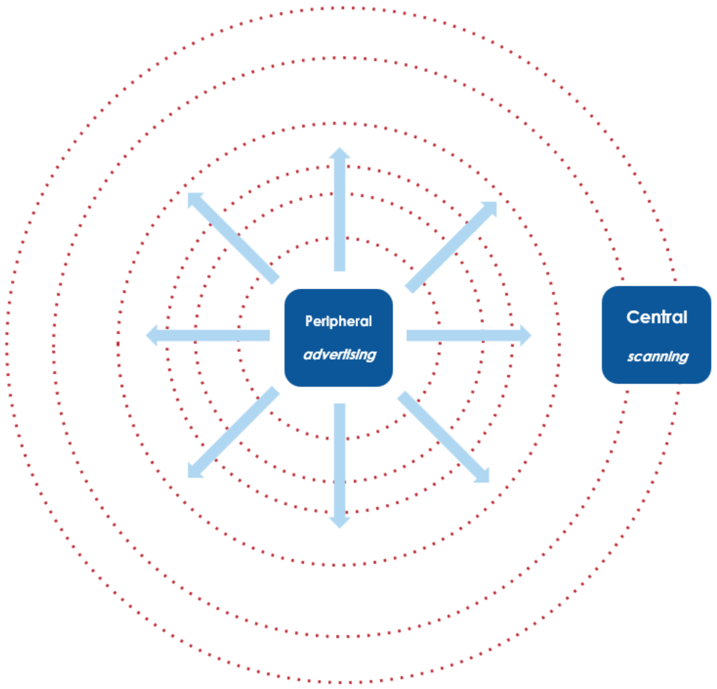


Figure 3 Peripheral Advertising and Central Scanning

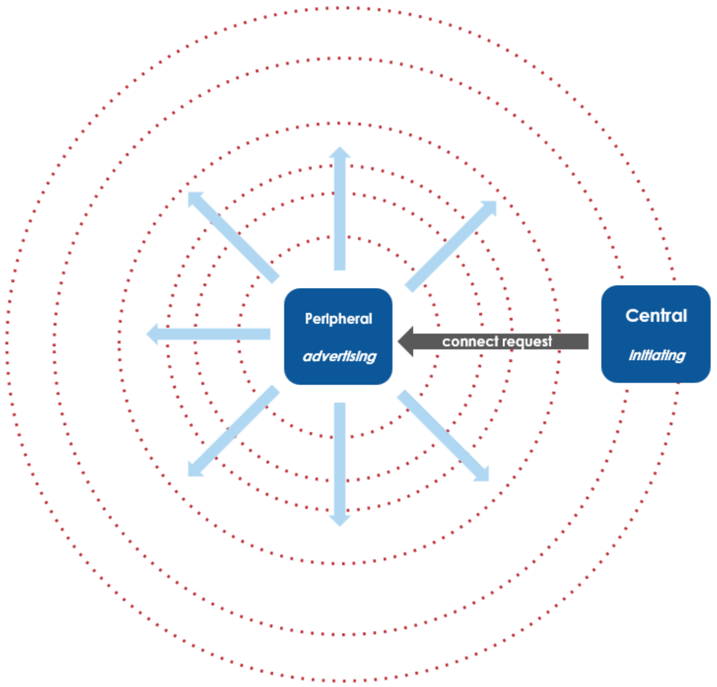


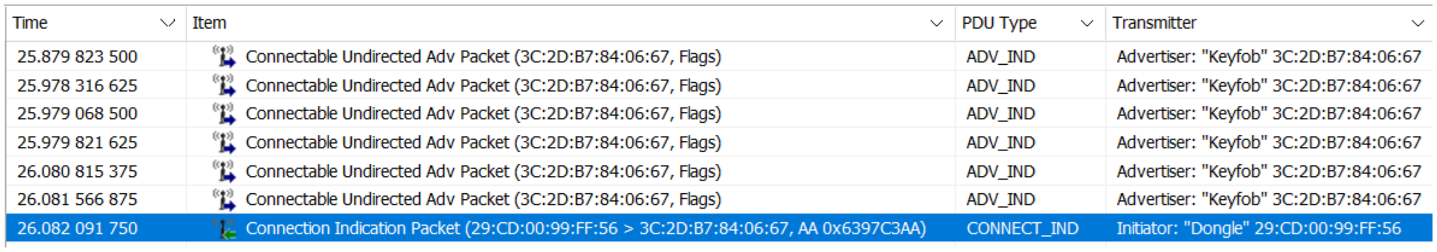
Figure 4   Central Sends a Connection Request PDU to Peripheral

Peripheral
connection
connected
Central
connection

Figure 5 The Devices are Now Connected

Note: Details such as the PDU transmitted to initiate the connection, and the radio channels used vary depending on whether the Peripheral is using legacy advertising ($ADVB_L$) or extended advertising ($ADVB_E$).

Below in Figure 6, the Ellisys Bluetooth Protocol Analysis software shows the transition from the advertising packets sent by the peripheral ("Advertiser") to the connection request from the Central ("Initiator"). The connection request is highlighted in blue.



Time	Item	PDU Type	Transmitter
25.879 823 500	Connectable Undirected Adv Packet (3C:2D:B7:84:06:67, Flags)	ADV_IND	Advertiser: "Keyfob" 3C:2D:B7:84:06:67
25.978 316 625	Connectable Undirected Adv Packet (3C:2D:B7:84:06:67, Flags)	ADV_IND	Advertiser: "Keyfob" 3C:2D:B7:84:06:67
25.979 068 500	Connectable Undirected Adv Packet (3C:2D:B7:84:06:67, Flags)	ADV_IND	Advertiser: "Keyfob" 3C:2D:B7:84:06:67
25.979 821 625	Connectable Undirected Adv Packet (3C:2D:B7:84:06:67, Flags)	ADV_IND	Advertiser: "Keyfob" 3C:2D:B7:84:06:67
26.080 815 375	Connectable Undirected Adv Packet (3C:2D:B7:84:06:67, Flags)	ADV_IND	Advertiser: "Keyfob" 3C:2D:B7:84:06:67
26.081 566 875	Connectable Undirected Adv Packet (3C:2D:B7:84:06:67, Flags)	ADV_IND	Advertiser: "Keyfob" 3C:2D:B7:84:06:67
26.082 091 750	Connection Indication Packet (29:CD:00:99:FF:56 > 3C:2D:B7:84:06:67, AA 0x6397C3AA)	CONNECT_IND	Initiator: "Dongle" 29:CD:00:99:FF:56

Figure 6 Advertising Packets Followed by Connection Request

Scheduling

A device usually has only a single radio, and that radio might need to be used for several different purposes at the same time (loosely speaking). For example, a device might have several active connections to several different devices, or it might be engaged in advertising at the same time as having a few LE-ACL connections in place, as shown in Figure 7, below. The radio can only do one thing at a time. It can be idle, it can be transmitting a packet, or it can be listening in receive mode. Consequently, the device's single radio must be shared across different concurrent uses, therefore the specification for each of the Bluetooth communication modes includes the definition of a radio **time-sharing** scheme.

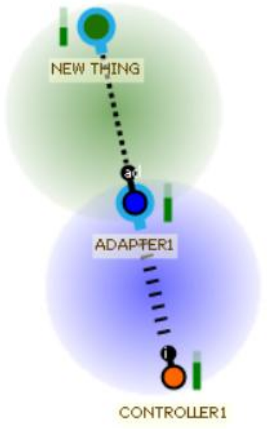
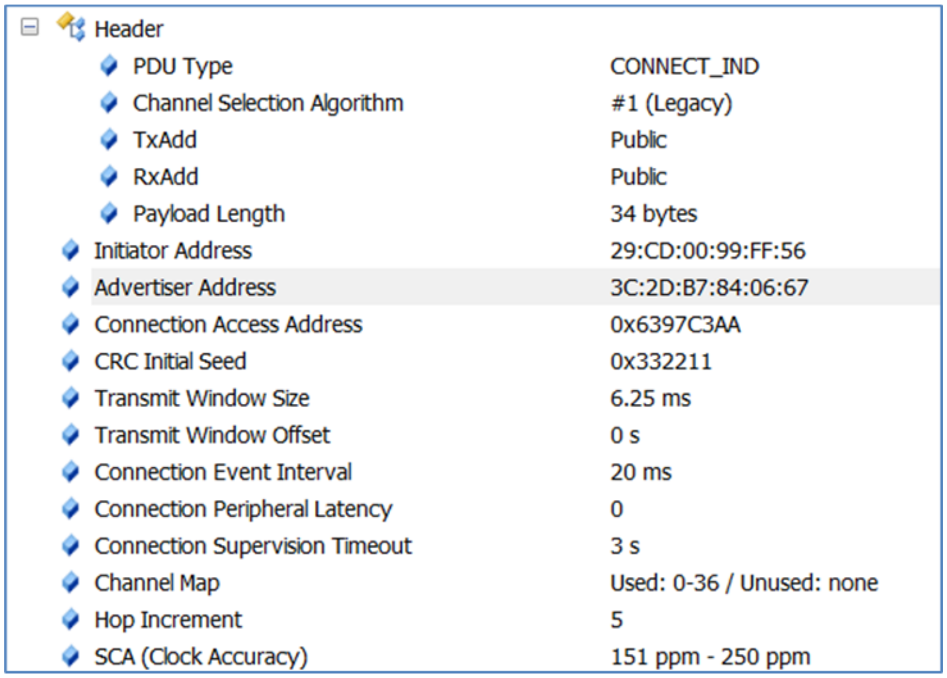


Figure 7 One Device (ADAPTER1) with an Established Connection as Peripheral (dotted line) and Concurrently Advertising (hashed line)

The connection request (CONNECT_IND PDU) which the Central device sends contains several parameters which dictate the timing and patterns of radio activity to be exhibited by the two devices, once connected (see Figure 8 below). Deciding when a connection will have the chance to use the radio is known as *scheduling*. Some aspects of scheduling are defined by the Bluetooth Core Specification and controlled by various standard parameters. Other aspects are left to the implementer.

Time sharing in an LE-ACL connection works along the following lines. A connection gets its opportunity to use the radio at regular and precisely timed intervals, the duration of which is set by a configuration parameter called the *connection interval*. Every time that interval comes around, a *connection event* is said to start, and the Central and

Peripheral commence taking turns to use the radio for the transmission of packets. The connection interval has a value that lies between 7.5 milliseconds and 4 seconds inclusive. Below in Figure 8 , the connection interval in a CONNECT_IND PDU is set to 20ms.



Header	
PDU Type	CONNECT_IND
Channel Selection Algorithm	#1 (Legacy)
TxAdd	Public
RxAdd	Public
Payload Length	34 bytes
Initiator Address	29:CD:00:99:FF:56
Advertiser Address	3C:2D:B7:84:06:67
Connection Access Address	0x6397C3AA
CRC Initial Seed	0x332211
Transmit Window Size	6.25 ms
Transmit Window Offset	0 s
Connection Event Interval	20 ms
Connection Peripheral Latency	0
Connection Supervision Timeout	3 s
Channel Map	Used: 0-36 / Unused: none
Hop Increment	5
SCA (Clock Accuracy)	151 ppm - 250 ppm

Figure 8 CONNECT_IND PDU Shown in Details View of the Ellisys Bluetooth Analysis Application

At the very start of a connection event, the Central should transmit a Link Layer data packet. The Peripheral, operating under the same set of timing parameters, will receive the packet and after a short delay period known as the inter-frame space (T_IFS), which defaults to 150 µs, will then transmit its own Link Layer data packet back to the Central. Note that there is no need for the content of the two data packets to be related in any way. But by taking turns transmitting, each device has a means of transferring data from one to the other.

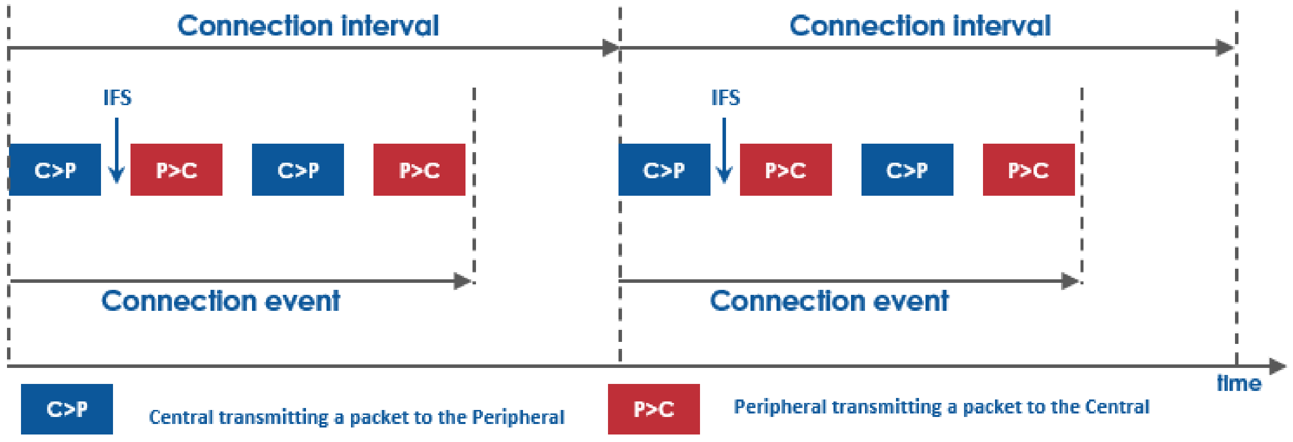


Figure 9 Connection Intervals and Connection Events in LE-ACL

There's an important point shown in Figure 9, above. Packets do not flow continually backwards and forwards across a connection without interruption from the moment the connection is established to the moment it is terminated. Packets are only exchanged during a connection event which starts every *connection interval* milliseconds. But a connection event is rarely the same length as the whole connection interval and so this leaves periods of time where the connection is not actively exchanging packets. It's these gaps that allow the radio to be scheduled for use for other purposes, such as by other connections or for advertising.

The figures below show a simple packet exchange in each direction (with T_IFS = 150 us), and a typical connection interval (7.5ms in this case).

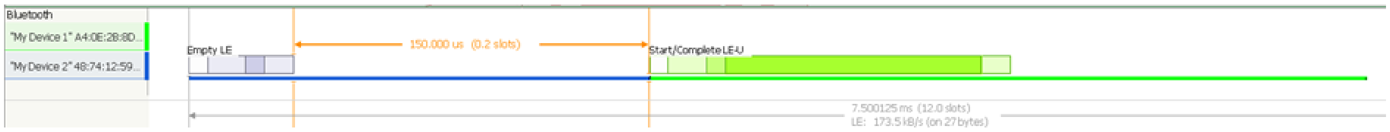

Bluetooth
"My Device 1" A4:0E:2B:8D...
"My Device 2" 48:74:12:59...
Empty LE
150.000 us (0.2 slots)
Start/Complete LE-U
7.500125 ms (12.0 slots)
LE: 173.5 kB/s (on 27 bytes)

Figure 10 Timing View from the Ellisys Analyzer Software. Packets Exchanged Between Central (Blue Underlined) and Peripheral (Green Underlined), Showing T_IFS = 150us.

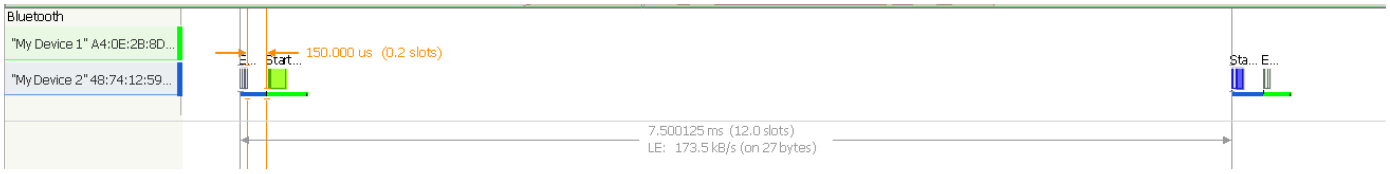
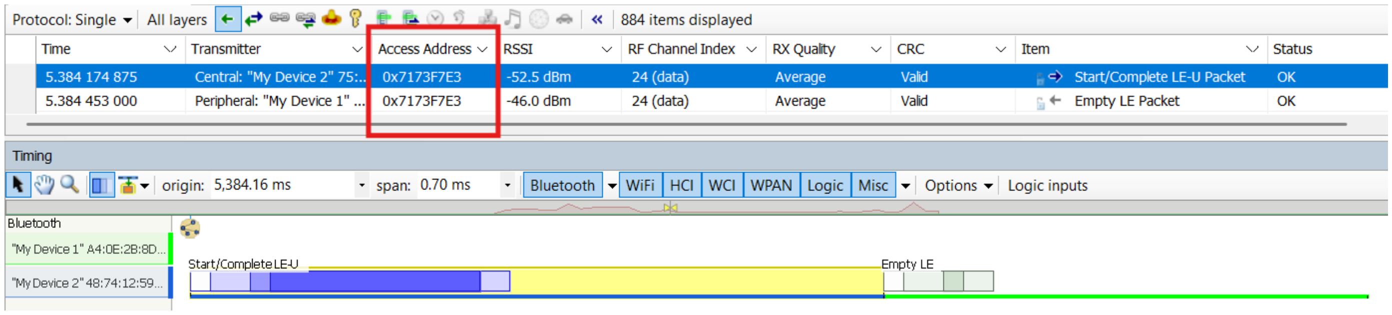

Bluetooth
"My Device 1" A4:0E:2B:8D...
"My Device 2" 48:74:12:59...
E...
Start...
150.000 us (0.2 slots)
Sta...
E...
7.500125 ms (12.0 slots)
LE: 173.5 kB/s (on 27 bytes)

Figure 11 Zoomed Out to Show a 7.5ms Connection Interval.

Addressing

When packets are transmitted over an LE-ACL connection, they always contain a link access address, as shown in Figure 12. This ensures other devices in range know that these packets are not intended for them. It's important to understand that this is not about security. Any device whose radio is tuned into the relevant channel can receive any Bluetooth packet. The inclusion of this address is just part of how an LE-ACL connection works; security is dealt with in other ways.

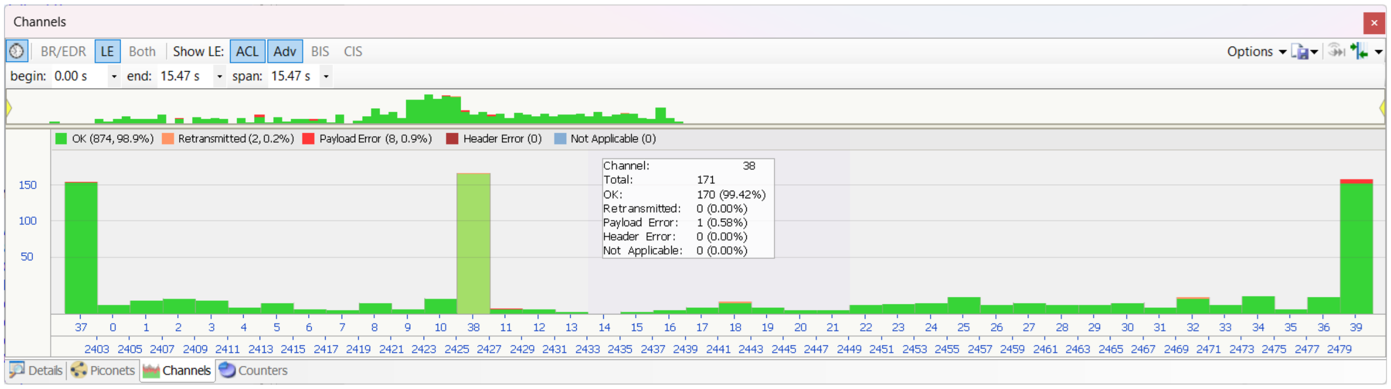


Time	Transmitter	Access Address	RSSI	RF Channel Index	RX Quality	CRC	Item	Status
5.384 174 875	Central: "My Device 2" 75:...	0x7173F7E3	-52.5 dBm	24 (data)	Average	Valid	Start/Complete LE-U Packet	OK
5.384 453 000	Peripheral: "My Device 1" ...	0x7173F7E3	-46.0 dBm	24 (data)	Average	Valid	Empty LE Packet	OK

Figure 12 Ellisys Bluetooth Analyzer Capture Showing Link Access Address used by Two Connected Devices

Radio Channels

In almost all of the communication modes, Bluetooth LE divides the 2.4 GHz band into 40 channels, each 2 MHz wide, as shown in Figure 13. Three of them are used for advertising, which leaves 37 to be used for communication modes such as LE-ACL. These 37 channels are known as the *general-purpose channels*.



```
Channel:                38
Total:             171
OK:                170 (99.42%)
Retransmitted:     0 (0.00%)
Payload  Error:    1 (0.58%)
Header  Error:     0 (0.00%)
Not  Applicable:   0 (0.00%)
```

Figure 13 General Purpose Channels (LE-ACL, 37 Channels Total) and Primary Advertising Channels (3 Channels Total) with the Channel Index Number and Frequency scales in view. The Advertising Channels are Indicated, in this Case, by the Three Higher Counts.

A channel is selected for use in each connection event using a *channel selection algorithm,* of which the Bluetooth Core Specification defines several. In all but the oldest of devices, an LE-ACL connection uses Channel Selection Algorithm #2 (CSA2).

CSA2 involves a pseudo-random number sequence that is seeded in the same way by both devices so that the same channel number is selected by both of them each time the algorithm is invoked. The overall effect of CSA2 is that over a sufficiently large number of connection events, no single channel is used significantly more than any other.

In any environment, there's always the chance that some of the general-purpose channels will not function well, perhaps being subject to interference from other devices. Bluetooth LE devices can use implementation-specific techniques to measure how well each channel is performing and if necessary, flag a badly performing channel so that it is no longer considered for use by the CSA2 algorithm. This process is informally known as *adaptive frequency hopping (AFH)*.

Devices keep track of the status of each of the general-purpose channels in a data structure called the *channel map*. Channels in the map are marked to indicate that they can be used or that they should not be used. The channel map's contents will change over time as environmental conditions change. Channels that start to perform badly get marked as *unused* in the channel map and channels that stop being subject to interference are made available for selection again by marking them as *used* in the channel map. The Bluetooth Core Specification states that a minimum of 2 channels must be available.

In *Figure 14* below, channel use by an LE-ACL connection is shown in the Channels view of Ellisys Bluetooth Analyzer software. Over the span selected for analysis (28.69s) we see heavy channel avoidance (magenta bars), mostly around the three non-overlapping 2.4G Wi-Fi channels (1, 6, and 11). Note the effectiveness of the AFH scheme, which results in a high-quality communications link and capture: 98.9% of the packets received as OK at the analyzer (thus easing debug tasks), and importantly, the retransmission rate is only 0.1%, indicating a clean, nearly error-free channel between the devices.

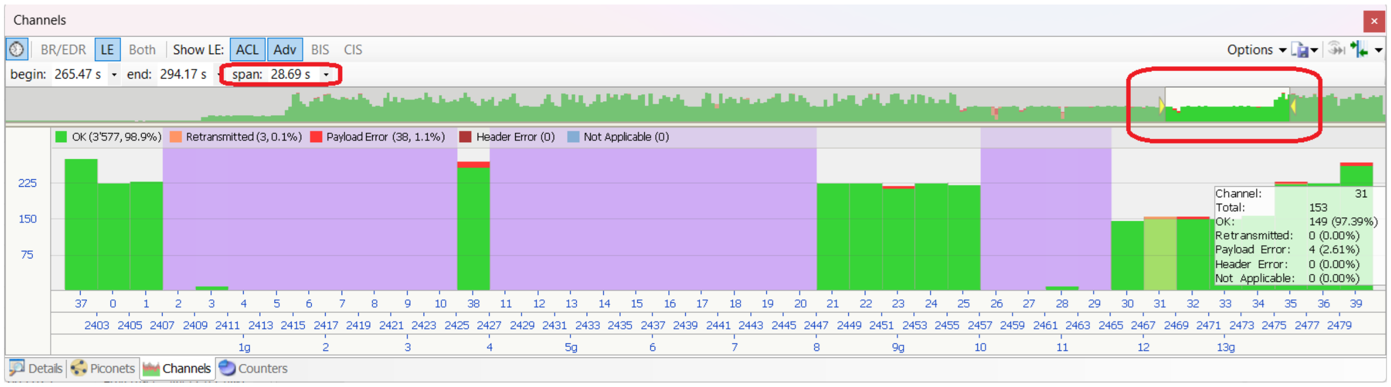


Figure 14 AFH Avoiding Interferences Generally Around Wi-Fi Channels 1, 6, and 11.

AFH provides a *major* benefit: the dynamic maintenance and use of information about channel performance means that devices can avoid problematic Bluetooth channels and continue to communicate effectively even when interference starts to be encountered in parts of the 2.4 GHz radio band. This significantly enhances reliability in Bluetooth LE connection-oriented communication.

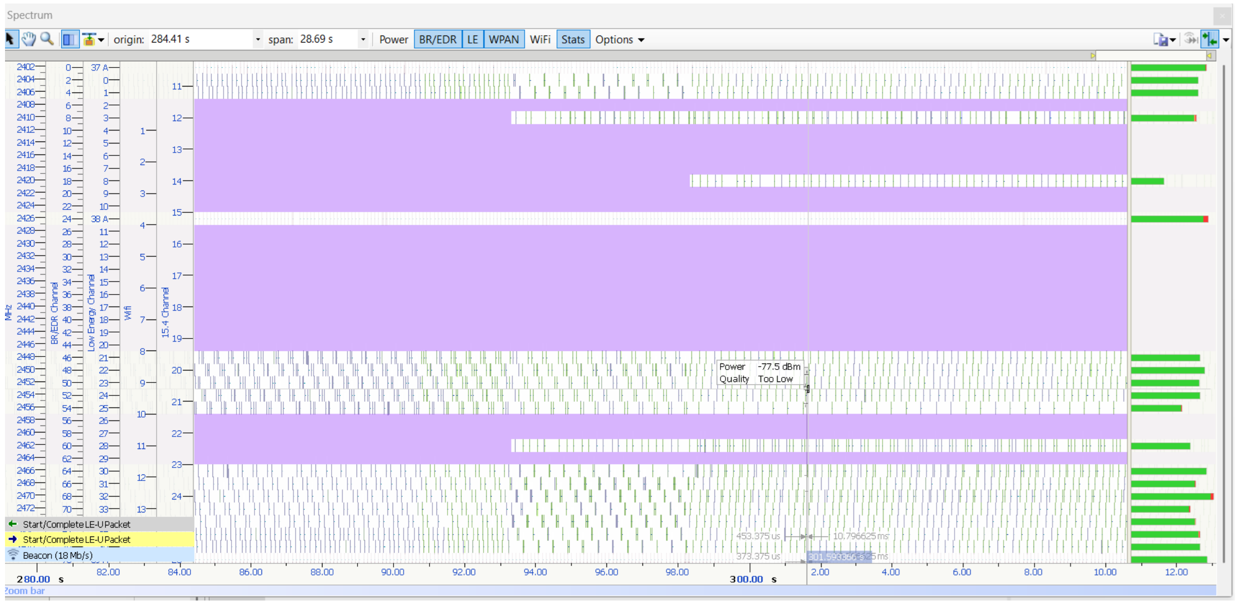


Figure 15 AFH is a Dynamic Approach to Managing Interferences (Ellisys Spectrum View).

The use of AFH brings with it one or two potential regulatory issues which need to be borne in mind:

- Regulators may have their own definitions of *adaptivity* which differ from the definition of *adaptive frequency hopping* as used with Bluetooth LE.
- Channel classification tends to reduce the number of general-purpose channels that are available for use. This increases the channel occupancy figure for each remaining "used" channel. Regulators usually have requirements relating to channel occupancy.

The Physical Layer

LE-ACL is a feature of the Link Layer in the Bluetooth LE stack and core specification.

Immediately below the Link Layer is the Physical Layer and amongst other things, it defines a series of configurations known as *PHYs*. One of the primary ways in which the various PHYs differ from each other is in the *symbol rate* at which they operate. Note that symbol rate is a measure of the rate of transmission of information in the analog world of radio and in the case of Bluetooth LE is directly comparable to *bit rate* in the digital world.

An LE-ACL connection may use one of three PHYs as summarized in the following table.

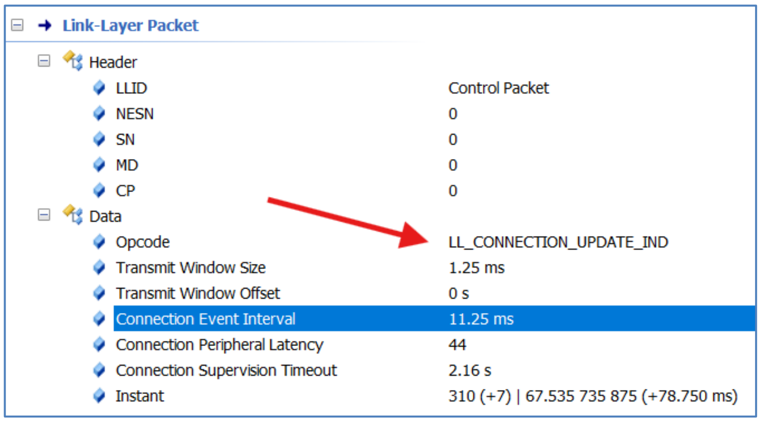
PHY Name	Symbol Rate	Error Correction?
LE 1M	1 M/s i.e., one million symbols transmitted per second.	NO
LE 2M	2 M/s i.e., two million symbols transmitted per second.	NO
LE Coded	1 M/s i.e., one million symbols transmitted per second.	YES

Table 1 LE-ACL PHY options

Packets, PDUs and Payloads

The Link Layer (LL) defines several packet formats for transmission over the air. Inside each such packet is a PDU (Protocol Data Unit), which in the case of LE-ACL logical transport will either be an *LL Control PDU* or an *LL Data PDU.*

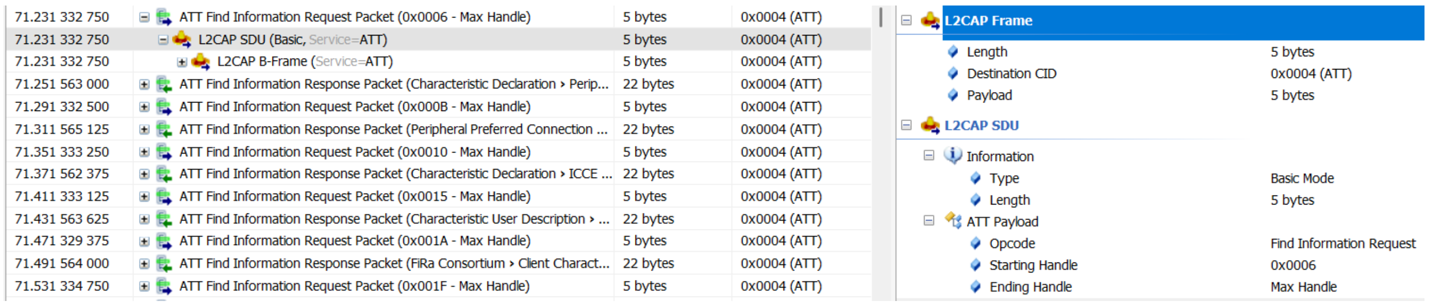
There are over fifty different types of LL Control PDUs and as indicated by the name, their purpose is to allow the link to be controlled by the two peer devices in various ways.



```
Link-Layer Packet
  Header
    LLID                              Control Packet
    NESN                              0
    SN                                0
    MD                                0
    CP                                0
  Data
    Opcode                            LL_CONNECTION_UPDATE_IND
    Transmit Window Size              1.25 ms
    Transmit Window Offset            0 s
    Connection Event Interval         11.25 ms
    Connection Peripheral Latency     44
    Connection Supervision Timeout    2.16 s
    Instant                           310 (+7) | 67.535 735 875 (+78.750 ms)
```

Figure 16 Link Layer Packet with a Control PDU (Connection Update).

LL Data PDUs carry the PDU from the layer of the stack that is immediately above the Link Layer as their payload. This is the Logical Link Control and Adaptation (L2CAP) layer. One of the responsibilities of L2CAP is *protocol multiplexing* and so the L2CAP PDU that a LL Data PDU contains might itself be a container for PDUs from other protocols that are defined even higher in the protocol stack. Common examples of what an LL DATA PDU might be transporting inside an L2CAP PDU are Attribute Protocol (ATT) PDUs or Security Manager Protocol (SMP) PDUs. So, LE-ACL PDUs contain PDUs which themselves contain other higher layer PDUs which ultimately, may contain application layer data.



71.231 332 750	ATT Find Information Request Packet (0x0006 - Max Handle)	5 bytes	0x0004 (ATT)
71.231 332 750	L2CAP SDU (Basic, Service=ATT)	5 bytes	0x0004 (ATT)
71.231 332 750	L2CAP B-Frame (Service=ATT)	5 bytes	0x0004 (ATT)
71.251 563 000	ATT Find Information Response Packet (Characteristic Declaration › Perip...	22 bytes	0x0004 (ATT)
71.291 332 500	ATT Find Information Request Packet (0x000B - Max Handle)	5 bytes	0x0004 (ATT)
71.311 565 125	ATT Find Information Response Packet (Peripheral Preferred Connection ...	22 bytes	0x0004 (ATT)
71.351 333 250	ATT Find Information Request Packet (0x0010 - Max Handle)	5 bytes	0x0004 (ATT)
71.371 562 375	ATT Find Information Response Packet (Characteristic Declaration › ICCE ...	22 bytes	0x0004 (ATT)
71.411 333 125	ATT Find Information Request Packet (0x0015 - Max Handle)	5 bytes	0x0004 (ATT)
71.431 563 625	ATT Find Information Response Packet (Characteristic User Description › ...	22 bytes	0x0004 (ATT)
71.471 329 375	ATT Find Information Request Packet (0x001A - Max Handle)	5 bytes	0x0004 (ATT)
71.491 564 000	ATT Find Information Response Packet (FiRa Consortium › Client Charact...	22 bytes	0x0004 (ATT)
71.531 334 750	ATT Find Information Request Packet (0x001F - Max Handle)	5 bytes	0x0004 (ATT)

L2CAP Frame	
Length	5 bytes
Destination CID	0x0004 (ATT)
Payload	5 bytes
L2CAP SDU	
Information	
Type	Basic Mode
Length	5 bytes
ATT Payload	
Opcode	Find Information Request
Starting Handle	0x0006
Ending Handle	Max Handle

Figure 17 An L2CAP PDU Containing an ATT Command (Find Information Request)

Connection Termination

Connections will usually eventually be terminated, and this can occur for a number of different reasons, such as:

- The application decides that the connection is no longer required. It can communicate this to the local Link Layer and a control PDU (LL_TERMINATE_IND) gets sent to the remote device to indicate that the connection is being closed.
- One of the parameters that the Central sends to the Peripheral when first setting up the connection is the *supervision timeout*. This indicates the maximum time which may elapse without receiving an LL Data PDU. If that time is exceeded, the connection is closed.

Application Concerns

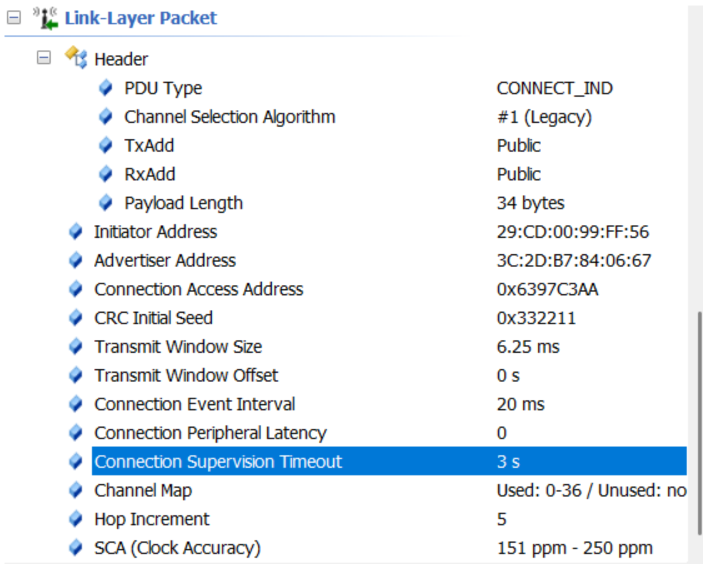
The part of a product that makes use of the Bluetooth LE stack is generally known as *the application*.

An application, in the sense that is meant here, can take many forms. It might be a consumer electronics product, or it might be a smartphone *app* that runs on iOS or Android. In either case, there's a clear distinction to be made between the parts that constitute the Bluetooth LE stack itself, those parts that make use of the stack, and those which have nothing whatsoever to do with communication, such as a graphical user interface.

Applications have their own set of requirements, some of which relate to wireless communication using Bluetooth LE and some of which do not. In this section, we'll consider a selection of those application issues that have a bearing on Bluetooth LE-ACL connections, how they can be used and how their inherent capabilities might help.

Configuration

There are a number of configuration parameters that affect the way an LE-ACL connection behaves, including the *connection interval* and the *supervision timeout*.



Link-Layer Packet

Header	
PDU Type	CONNECT_IND
Channel Selection Algorithm	#1 (Legacy)
TxAdd	Public
RxAdd	Public
Payload Length	34 bytes
Initiator Address	29:CD:00:99:FF:56
Advertiser Address	3C:2D:B7:84:06:67
Connection Access Address	0x6397C3AA
CRC Initial Seed	0x332211
Transmit Window Size	6.25 ms
Transmit Window Offset	0 s
Connection Event Interval	20 ms
Connection Peripheral Latency	0
Connection Supervision Timeout	3 s
Channel Map	Used: 0-36 / Unused: no
Hop Increment	5
SCA (Clock Accuracy)	151 ppm - 250 ppm

Figure 18 Connection Parameters Asserted at Connection in the CONNECT_IND PDU, including Connection Supervision Timeout and Connection Interval.

The Bluetooth LE stack has a series of layers which are grouped within two major parts within the overall architecture, called the *Host* and the *Controller*. The Host is generally an operating system, and the Controller is generally a chip, but it doesn't have to be this way. Host and Controller are logical constructs which allow for the physical separation of the higher and lower layers of the stack rather than strict implementation rules.

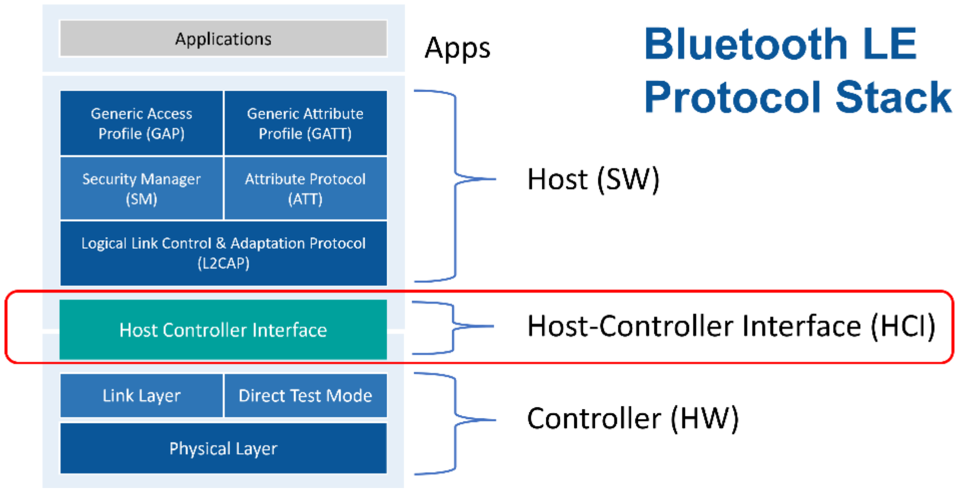

Applications
Apps
Bluetooth LE Protocol Stack
Generic Access Profile (GAP)
Generic Attribute Profile (GATT)
Security Manager (SM)
Attribute Protocol (ATT)
Logical Link Control & Adaptation Protocol (L2CAP)
Host (SW)
Host Controller Interface
Host-Controller Interface (HCI)
Link Layer
Direct Test Mode
Physical Layer
Controller (HW)

Figure 19 Bluetooth LE Stack

Sitting in between the Host and Controller is a layer called the Host Controller Interface (HCI). Its purpose is to allow the Host to send *commands* to the Controller and for the Controller to send data structures called *events* to the Host. Events might contain the response to a command, or they might contain data which the Controller needs to communicate to the Host independently.

HCI provides the mechanism by which an application *might* be able to configure the key parameters of an LE-ACL connection. HCI defines the *HCI_LE_Create_Connection* command which includes minimum and maximum connection interval parameters and a supervision timeout parameter.

However, an application will typically not be able to use HCI commands and events directly. An Android application for example, cannot formulate and submit HCI commands straight to the Bluetooth LE controller in the smartphone it is running on. Instead, application developers use collections of functions called *APIs* (Application Programming Interface) in their code. APIs are provided by the platform and/or software development kit (SDK) that the developer is working with and typically make available only a subset of HCI capabilities to the application developer. And of those HCI features that are available, APIs will often impose their own rules which constrain the application developer in some way.

For example, as was already mentioned, the connection interval parameter of an LE-ACL connection has a defined range of 7.5 ms to 4 seconds. But an Android developer cannot specify a connection interval when initiating a connection and the value that this parameter is assigned in the stack is determined by the operating system. The same is true of the supervision timeout parameter.

On the other hand, the Android APIs do allow the application developer to request the use of one of the three supported PHY types for a connection.

The lesson here is that application developers shouldn't assume that everything defined in the Bluetooth Core Specification HCI section is available for them to use. In some cases, APIs expose a feature in full, in others an API might provide access to (say) a certain command but with restrictions placed on parameter values, and in other cases it might be that there is no access to an HCI feature provided by the 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 target platform.

It should also be noted that the configuration that the Central requests when initiating the connection is not necessarily retained for the full lifespan of the connection. Both the Central and the Peripheral can later request that parameters be changed using a Link Layer procedure called the Connection Parameters Request procedure.

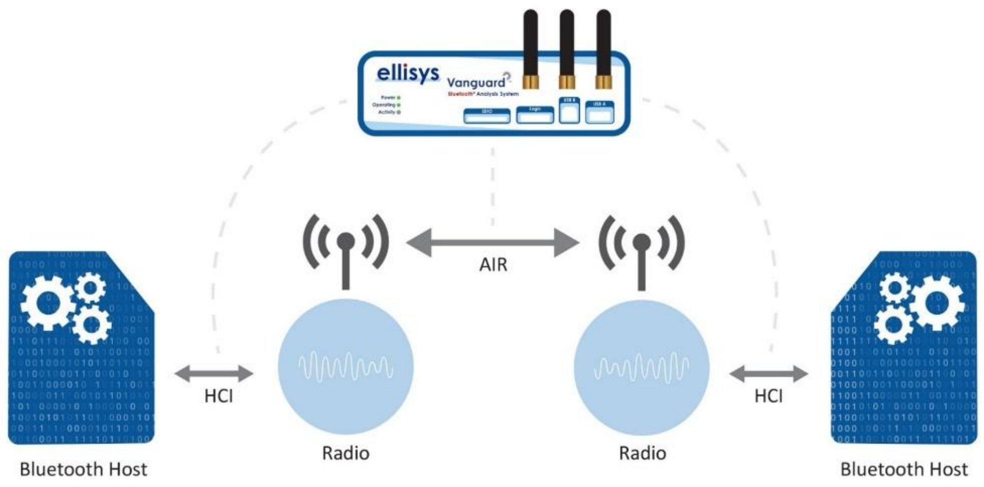


Figure 20 Ellisys Bluetooth Vanguard Analyzer Capturing Over-the-Air Traffic and HCI traffic on Both Sides of a Connection

Energy Efficiency

Wasting energy and reducing battery life is never a good thing but sometimes energy efficiency, especially in smaller Peripheral devices, is of paramount importance. It's not uncommon to encounter a requirement that stipulates that a device must be able to run off a coin cell battery for a number of years.

LE-ACL connections offer a number of ways in which energy can be used more sparingly:

- **Peripheral Latency** - this connection parameter allows the Peripheral to *not listen* for packets from the Central over a number of consecutive connection events. This results in energy being saved by the Peripheral through not having its radio in receive mode for every time slot where the Central transmits and through not needing to transmit packets in response. The Central can indicate a maximum peripheral latency value in the HCI_LE_Create_Connection command, but the Peripheral can request a different value using the Connection Parameters Request procedure. In this way, a Peripheral may be able to optimize this aspect of the connection's behavior to better meet its requirements.

- **Subrating** - A subrated connection is one where a subset of connection events is used for radio activity by either the Central device or the Peripheral. In some respects, subrating provides the Central device with a similar energy saving mechanism to that which *peripheral latency* provides to the Peripheral device. Subrating is about more than just energy efficiency, however. It allows the connection to be in one of two states. The first might be likened to a standby mode with the connection established and requiring very little energy to maintain it but providing a slow data transfer rate and exhibiting long latency times. The second state provides low latency and higher performance. Best of all, subrating allows a device to switch between the two states instantly. This is good for applications whose communication requirements can change quickly and unpredictably.

- **LE Power Control** - This feature allows devices to exercise dynamic control of transmission power levels. Applications can set parameters via HCI commands, and the local and remote devices then communicate, reporting local measurements such as path loss and making requests to the remote device to change its transmission power level. Optimization of transmission power levels to suit local conditions can sometimes improve energy efficiency.

- **Connection Interval** - A larger connection interval value might reduce energy use but is not guaranteed to do so. It all depends on how long the connection events are within the connection interval, i.e., how many packets are transmitted and received, because it is the use of the radio that is the key consumer of energy in this context.

Reliability

Radio communication is not inherently reliable. Well-designed protocols and procedures can make communication reliable though, even given that the underlying foundations are not.

LE-ACL connections have a number of aspects of their design that make them a good choice for those occasions where a more reliable transport is needed:

- **Coping with Interference.** In most environments, there is the risk of interference from other sources of radio transmissions. But LE-ACL connections will continue to work well even when within range of numerous other transmitting devices. This is because adaptive frequency hopping distributes transmissions randomly across up to 37 different radio channels, and the odds of collisions taking place with the transmissions from other devices are very much reduced. And any channel that is found to be experiencing interference can be removed from the channel map so that the problematic channel is avoided.

- **Arrival and Ordering**. The most fundamental requirement for any packet transmitted over a connection is that it must be received. Packets also need to be received in the intended order. LE-ACL uses three single bit fields called the Sequence Number (SN), Next Expected Sequence Number (NESN) and More Data (MD) fields (located in the packet's header) to implement a system for detecting packets received out of order for and for acknowledging to the transmitting device that a packet was received.

- **Error Detection.** The effects of interference can be detected by a receiving device. Every packet includes a Cyclic Redundancy Check (CRC) field whose value is calculated from the values of other fields by the transmitting device. The receiving device calculates the expected CRC value in the same way and if the value in the received packet does not match the locally calculated value, data corruption due to interference has probably taken place. By not acknowledging a packet whose CRC check fails, the receiver can indicate to the transmitter that it should send the packet again.

Security

There are many aspects to the subject of security. Amongst the most fundamental requirements that an application often has are:

- **Confidentiality**: Data must remain confidential whilst in flight and it must not be possible for a third party to eavesdrop on any exchange of data and extract meaningful information from its content.

- **Authentication**: It must not be possible for a 3rd party to impersonate a trusted device.

- **Tamper Detection**: It must not be possible for a 3rd party to modify data in flight without this being detected by the intended recipient.

LE-ACL connections can be protected by Link Layer encryption. When this feature is enabled, data PDUs are encrypted, and this provides protection against passive eavesdropping.

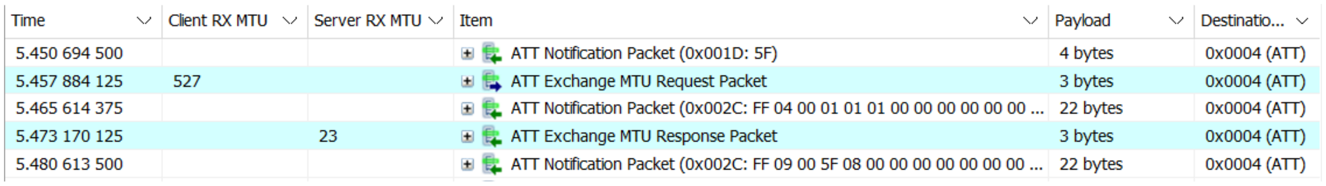
Encrypted PDUs include an extra field called the Message Integrity Check (MIC) field. The MIC allows the receiving device to *authenticate* the received packet and verify that it came from the trusted device that it is in a connection with rather than some other malicious device that is attempting to forge packets. It also allows for the detection of changes made to packets by devices other than the trusted originator.

For Link Layer encryption and PDU authentication to be available for use, the two connected devices must have been paired. Pairing generates and distributes key material which can be used to create various security keys that are used by the Link Layer security system.

Maximizing Data Throughput Rates

For some applications, transferring application data between devices as quickly as possible is a key requirement. The rate of transfer of application data will always be less than the symbol rate of the selected PHY because a proportion of the transmitted data is protocol data rather than application data, so only part of the overall bandwidth is available to applications. Fortunately, however, there are some variables which applications may be able to set or influence that can help achieve higher application data transfer rates:

- **Maximum Transmission Unit (MTU)** - This parameter sets the maximum number of bytes of application data that can be included in the PDUs of upper layer protocols like the Attribute Protocol which is very often used over an LE-ACL connection. As such, the MTU is also the upper limit for the amount of application data in each LL Data PDU transmitted over the air in packets. The larger the MTU value, the greater the proportion of available bandwidth used for application data and therefore the higher the application data transmission rate achieved. Applications can request higher MTU sizes if a suitable API is available.



Time	Client RX MTU	Server RX MTU	Item	Payload	Destinatio...
5.450 694 500			ATT Notification Packet (0x001D: 5F)	4 bytes	0x0004 (ATT)
5.457 884 125	527		ATT Exchange MTU Request Packet	3 bytes	0x0004 (ATT)
5.465 614 375			ATT Notification Packet (0x002C: FF 04 00 01 01 01 00 00 00 00 00 00 ...	22 bytes	0x0004 (ATT)
5.473 170 125		23	ATT Exchange MTU Response Packet	3 bytes	0x0004 (ATT)
5.480 613 500			ATT Notification Packet (0x002C: FF 09 00 5F 08 00 00 00 00 00 00 00 ...	22 bytes	0x0004 (ATT)

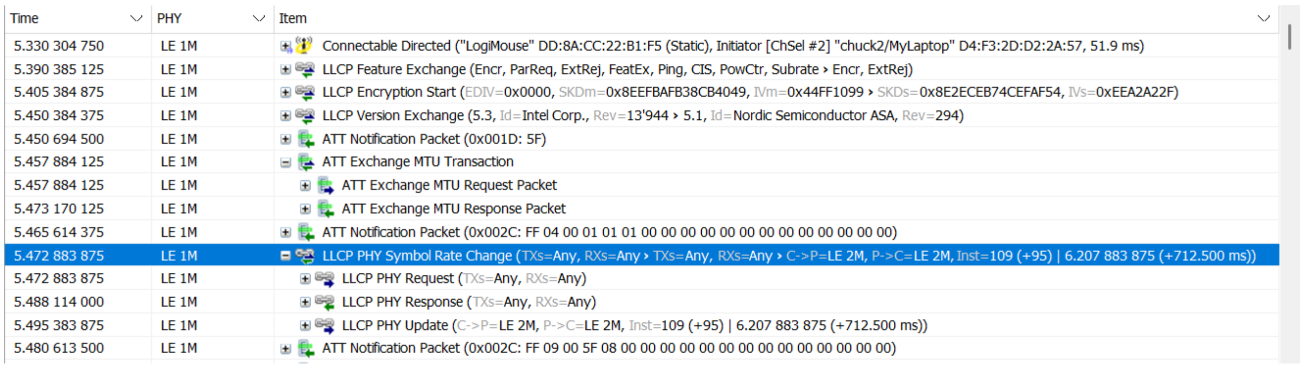
Figure 21 Typical Client-Server Exchange over Attribute Protocol for MTU Values

- **LE Data Packet Length Extension** - This Link Layer feature allows larger link layer packets to be used and therefore each packet can contain more application data with less overhead due to protocol data overall. If both devices support this feature its use will automatically be negotiated and applications will benefit without needing to take any special action.

- **Connection Interval**: The shorter the connection interval, the more often the connection used by an application will have the opportunity to transmit data. This might increase the overall application data transmission, but not necessarily. What matters is the number of packets transmitted in each connection event (all other variables being equal). But scheduling is a complex topic which is only partly dealt with by the Bluetooth Core Specification, largely being left to the implementation (e.g., the Android operating system) to address. Consequently, there's no guarantee that reducing the connection interval will increase the application data transmission rate. It's worth trying, however.

- **PHY and Symbol Rate**: The LE 2M PHY has a symbol rate that is double that of the LE 1M PHY. You'd quite reasonably expect LE 2M to provide a much faster application data transmission rate than LE 1M. But once again, this is not necessarily going to be true because there's another way of looking at the impact of LE 2M compared with LE 1M. *LE 2M saves time*. From the perspective of the implementer of Bluetooth scheduling in an operating system, say, the prime benefit of LE 2M is that the same amount of data can be transmitted in half the time and this leaves precious unused time which can be used by the scheduling algorithm for other purposes such as for other connections. But once again, it's worth trying LE 2M and comparing the results obtained with LE 1M.

In Figure 22 below, two devices have connected, exchanged various link layer information, then exchange PHY preferences and capabilities via ATT protocol. A bit later, the connection will progress to LE 2M as calculated by the analyzer software and indicated by the timestamp associated with the LLCP PHY Update (blue highlight).



Time	PHY	Item
5.330 304 750	LE 1M	Connectable Directed ("LogiMouse" DD:8A:CC:22:B1:F5 (Static), Initiator [ChSel #2] "chuck2/MyLaptop" D4:F3:2D:D2:2A:57, 51.9 ms)
5.390 385 125	LE 1M	LLCP Feature Exchange (Encr, ParReq, ExtRej, FeatEx, Ping, CIS, PowCtr, Subrate › Encr, ExtRej)
5.405 384 875	LE 1M	LLCP Encryption Start (EDIV=0x0000, SKDm=0x8EEFBAFB38CB4049, IVm=0x44FF1099 › SKDs=0x8E2ECEB74CEFAF54, IVs=0xEEA2A22F)
5.450 384 375	LE 1M	LLCP Version Exchange (5.3, Id=Intel Corp., Rev=13'944 › 5.1, Id=Nordic Semiconductor ASA, Rev=294)
5.450 694 500	LE 1M	ATT Notification Packet (0x001D: 5F)
5.457 884 125	LE 1M	ATT Exchange MTU Transaction
5.457 884 125	LE 1M	ATT Exchange MTU Request Packet
5.473 170 125	LE 1M	ATT Exchange MTU Response Packet
5.465 614 375	LE 1M	ATT Notification Packet (0x002C: FF 04 00 01 01 01 00 00 00 00 00 00 00 00 00 00 00 00 00)
5.472 883 875	LE 1M	LLCP PHY Symbol Rate Change (TXs=Any, RXs=Any › TXs=Any, RXs=Any › C->P=LE 2M, P->C=LE 2M, Inst=109 (+95) | 6.207 883 875 (+712.500 ms))
5.472 883 875	LE 1M	LLCP PHY Request (TXs=Any, RXs=Any)
5.488 114 000	LE 1M	LLCP PHY Response (TXs=Any, RXs=Any)
5.495 383 875	LE 1M	LLCP PHY Update (C->P=LE 2M, P->C=LE 2M, Inst=109 (+95) | 6.207 883 875 (+712.500 ms))
5.480 613 500	LE 1M	ATT Notification Packet (0x002C: FF 09 00 5F 08 00 00 00 00 00 00 00 00 00 00 00 00 00 00)

Figure 22 Two Devices Exchange PHY Capabilities and Preferences.

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. Repeated here is the table of properties for LE-ACL connections.

Property	Comment
Topology	One-to-one (1:1).
Transmitters vs Receivers	Devices take turns to transmit and receive.
Application Data Direction	Application data can be transmitted in both directions.
Connected or Connectionless?	Connected.
Data and Time	Asynchronous.
Receiver Concurrency	Data packets are addressed to one receiver device at a time.
Radio Channels	Uses 37 of 40 x 2 MHz wide channels. Channel selection involves a process known as adaptive frequency hopping.
Scalability	Some devices can handle establishing more than one connection to other devices at a time. The Bluetooth Core Specification does not define a limit, but implementation issues generally limit this to a fairly low number. Application data throughput over an LE-ACL connection can be significantly improved using certain configuration parameters but will always be less than the underlying PHY symbol rate.
Choice of PHY	LE 1M, LE 2M, or LE Coded.

Table 2 LE-ACL 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 LE-ACL connections.

Did you know that the number of packets to be exchanged in a connection event is not defined in the Bluetooth Core Specification

Transmitting a Link Layer data packet takes a short amount of time. Even the largest possible packet is only going to take about 2000 µs. The connection interval can be anything from 7.5 ms to 4 seconds. Usually there's plenty of time, *in theory,* for the Central device to exchange multiple packets with the Peripheral. In practice, there may be more at work in the scheduling process than meets the eye, and this can result in a small proportion of the connection interval used for the connection event and packet exchange.

Did you know that even though the Central would usually transmit a packet at the start of a connection event, it doesn't have to?

Usually, the Central will transmit a packet at the start of each connection event, but this is not mandatory. A scheduling conflict might prevent the Central from transmitting in that time slot, for example. The Central device must transmit at least one packet containing a LL Data PDU within the supervision period defined for the connection though. If it doesn't, the connection will be closed.

Did you know that LE-ACL provides a simple flow control mechanism?

Flow control is a technique used to manage the rate at which data is transferred between parts of a communication system. The main aim of flow control is to ensure that the part sending data does not exceed the arrival rate that the receiving part can handle, as this can result in data loss (e.g., due to buffer overflow). LE-ACL has a simple flow control mechanism which involves deliberately not updating the Next Expected Sequence Number field, as this causes the remote device to retransmit the same packet later and has the effect of slowing down the connection. A more sophisticated flow control mechanism is available in the L2CAP layer.

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

Until Bluetooth Core Specification version 6.0 was released, the inter-frame space time parameter which defines the time to wait in between the transmission of packets (amongst other things) was fixed at 150 µs. Version 6.0 changed this, and now devices can use different frame space values to separate each Central transmission and the subsequent time slot that the Peripheral could use and vice versa. Inter-frame space time values still default to 150 µs, but devices can negotiate new values in the range 0 to 10,000 µs.

Did you know that LE-ACL requires both devices to have a sufficiently accurate clock with which to time activities?

The Bluetooth Core Specification defines the clock accuracy requirements for devices and particular communication modes.

Did you know that CSA2 results in an even distribution of channel use over a large enough number of connection events?

On average, each channel is used for 2.7% of the time (assuming all 37 channels are available).

Did you know that some regional regulatory requirements have their own definition of adaptive vs non-adaptive channel selection behaviors?

It's important to check these definitions and understand how they relate to a product's use of Bluetooth LE. The Bluetooth SIG provides a document called the Regulatory Aspects Document (RAD) which can help with this and other regulatory questions.

Next in the Series

In this article we've explored the LE-ACL communication mode of Bluetooth LE.

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