Bluetooth

What is Bluetooth?

A Bluetooth technology is a high speed low powered wireless technology link that is designed to connect phones or other portable equipment together. It is a specification (IEEE 802.15.1) for the use of low power radio communications to link phones, computers and other network devices over short distances without wires. Wireless signals transmitted with Bluetooth cover short distances, typically up to 30 feet (10 meters).

It is achieved by embedded low cost transceivers into the devices. It supports on the frequency band of 2.45GHz and can support upto 721KBps along with three voice channels. This frequency band has been set aside by international agreement for the use of industrial, scientific and medical devices (ISM).rd-compatible with 1.0 devices.

Bluetooth can connect up to eight devices simultaneously and each device offers a unique 48 bit address from the IEEE 802 standard with the connections being made point to point or multipoint.

Working of Bluetooth

Bluetooth Network consists of a Personal Area Network or a piconet which contains a minimum of 2 to maximum of 8 bluetooth peer devices- Usually a single master and upto 7 slaves.

A master is the device which initiates communication with other devices. The master device governs the communications link and traffic between itself and the slave devices associated with it.

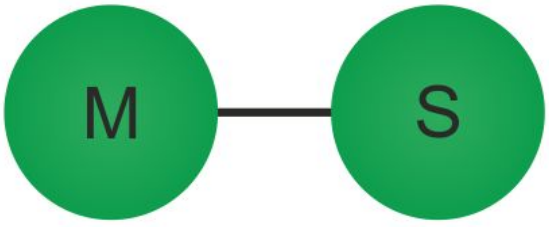
A slave device is the device that responds to the master device. Slave devices are required to synchronize their transmit/receive timing with that of the masters.

In addition, transmissions by slave devices are governed by the master device (i.e., the master device dictates when a slave device may transmit). Specifically, a slave may only begin its transmissions in a time slot immediately following the time slot in which it was addressed by the master, or in a time slot explicitly reserved for use by the slave device.

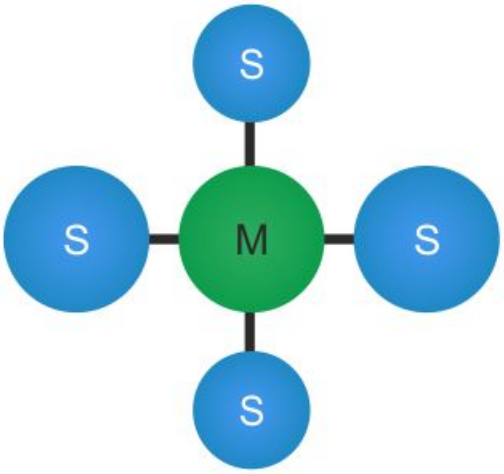
The frequency hopping sequence is defined by the Bluetooth device address (BD_ADDR) of the master device. The master device first sends a radio signal asking for response from the particular slave devices within the range of addresses. The slaves respond and synchronize their hop frequency as well as clock with that of the master device.

Scatternets are created when a device becomes an active member of more than one piconet. Essentially, the adjoining device shares its time slots among the different piconets.

Point-to-point link (Single master and single slave):



Point-to-multiple link (Single master and multiple slaves):

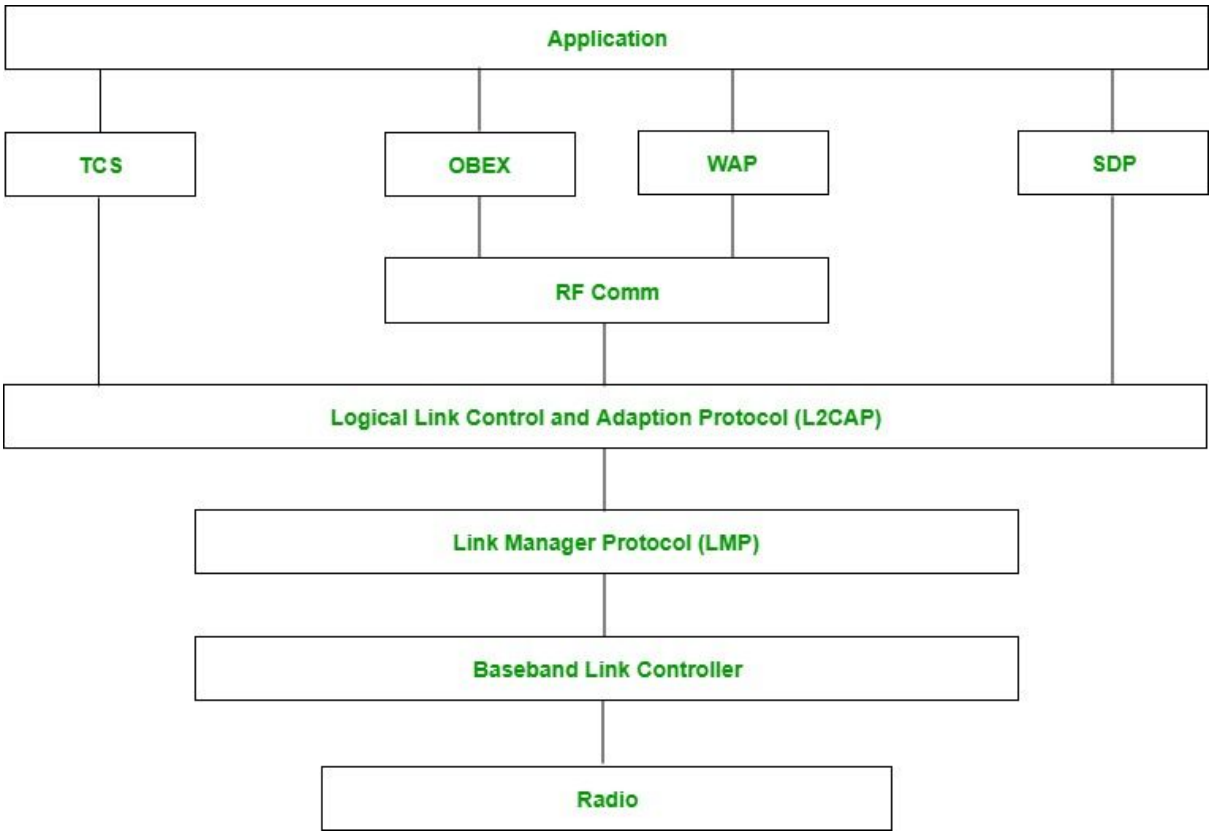


Bluetooth Addressing System

Every Bluetooth device has a unique 48-bit address, commonly abbreviated BD_ADDR. This will usually be presented in the form of a 12-digit hexadecimal value. The most-significant half (24 bits) of the address is an organization unique identifier (OUI), which identifies the manufacturer. The lower 24-bits are the more unique part of the address.

Bluetooth Protocol Stack

Bluetooth protocol stack defines and provides different types of layers and functionalities. Bluetooth can run the different applications over different protocol stacks, but each one of these protocol stacks uses the same Bluetooth link and physical layers. The below diagram shows a complete Bluetooth protocol stack. It shows the relationship between the protocols that use the services of other protocols when there is a payload to be transferred in the air.



Layers of Bluetooth Protocol Stack

1. Radio (RF) layer:

It performs modulation/demodulation of the data into RF signals. It defines the physical characteristics of a Bluetooth transceiver. It defines two types of physical link: connection-less and connection-oriented.

2. Baseband Link layer:

It performs the connection establishment within a piconet.

3. Link Manager protocol layer:

It performs the management of the already established links. It also includes authentication and encryption processes.

4. Logical Link Control and Adaptation protocol layer:

It is also known as the heart of the Bluetooth protocol stack. It allows the communication between upper and lower layers of the Bluetooth protocol stack. It packages the data packets received from upper layers into the form expected by lower layers. It also performs the segmentation and multiplexing.

5. SDP layer:

It is short for Service Discovery Protocol. It allows discovering the services available on another Bluetooth enabled device.

6. RF comm layer:

It is short for Radio Frontend Component. It provides a serial interface with WAP and OBEX.

7. OBEX:

It is short for Object Exchange. It is a communication protocol to exchange objects between 2 devices.

8. WAP:

It is short for Wireless Access Protocol. It is used for internet access.

9. TCS:

It is short for Telephony Control Protocol. It provides telephony service.

10. Application layer:

It enables the user to interact with the application.

Types of Protocols

1. RFCOMM

The RFCOMM protocol provides roughly the same service and reliability guarantees as TCP. Although the specification explicitly states that it was designed to emulate RS-232 serial ports (to make it easier for manufacturers to add Bluetooth capabilities to their existing serial port devices), it is quite simple to use it in many of the same scenarios as TCP.

Implementation

Server:

```
from bluetooth import *

port = 1

server_sock=BluetoothSocket( RFCOMM )
server_sock.bind(("",port))
server_sock.listen(1)

client_sock, client_info = server_sock.accept()
print("Accepted connection from ", client_info)

data = client_sock.recv(1024)
print("received [%s]" % data)

client_sock.close()
server_sock.close()
```

Client:

```
from bluetooth import *
server_address = "01:23:45:67:89:AB"
port = 1

sock=BluetoothSocket( RFCOMM )
sock.connect((server_address, port))

sock.send("hello")

sock.close()
```

2. L2CAP

UDP is often used in situations where reliable delivery of every packet is not crucial, and sometimes to avoid the additional overhead incurred by TCP. Specifically, UDP is chosen for its best-effort, simple datagram semantics. These are the same criteria that L2CAP satisfies as a communications protocol.

Implementation

Server:

```
from bluetooth import *

port = 0x1001

server_sock=BluetoothSocket( L2CAP )
server_sock.bind(("",port))
server_sock.listen(1)
```

```
client_sock,address = server_sock.accept()
print("Accepted connection from ",address)
data = client_sock.recv(1024)
print("received [%s]" % data)

client_sock.close()
server_sock.close()
```

Client:

```
from bluetooth import *

sock=BluetoothSocket(L2CAP)

bd_addr = "01:23:45:67:89:AB"
port = 0x1001

sock.connect((bd_addr, port))

sock.send("hello")

sock.close()
```