Chapter 5A: Classic Bluetooth – The Wireless Serial Port

Time: 3 Hours

At the end of this chapter you will understand the basics of Classic Bluetooth and how to create a simple Classic Bluetooth project on WICED devices. This section is focused on the simplest Bluetooth connection, one Master (Android, Mac or PC) and one Slave (your WICED Bluetooth Device). By the end you should understand Inquiry, Page, Pair, Bond, SDP and SPP.

5.1 WICED Bluetooth Classic System Lifecycle 2

5.1.1 Inquiry 3

5.1.2 Page / Connect 4

5.1.3 Pair & Bond 4

5.1.4 Discover the Services using Service Discovery Protocol (SDP) 4

5.1.5 Exchange Data with the Serial Port Profile 5

5.2 Secure Simple Pairing 5

5.3 Service Discovery Protocol (SDP) 5

5.4 L2CAP, RFCOMM & the Serial Port Profile 6

5.4.1 L2CAP 6

5.4.2 RFCOMM 6

5.4.3 Serial Port Profile 7

5.5 WICED Bluetooth Designer 7

5.6 WICED Classic Bluetooth Firmware Architecture 7

5.7 Exercises 7

# WICED Bluetooth Classic System Lifecycle

The Bluetooth Classic Spec has a bewildering amount of complexity. Clearly this must have been one of the motivations for creating the much simpler BLE standard. Like Chapter 4 we will take the approach of creating the simplest example project possible to get things going.

The simplest Bluetooth Classic scenario has two devices, a Master and a Slave. Slaves are passive – not transmitting – until they hear an Inquiry broadcast from a Master, at which point the Slave broadcasts basic information about itself (Name, BDADDR, Services). The Master then Pages (connects) to the Slave and they exchange and save Pairing information. The Master then discovers the Services i.e. capabilities of the Slave. Finally, a basic wireless Serial Port data exchange connection is created.

The five steps are:

1. Inquiry – Master finds a Slave to Connect
2. Paging – Master connects to Slave
3. Pair & Bond – A secure, authenticated connection is created
4. Service Discovery (SDP) – The Master figures out what the Slave can do
5. Exchange Data using the Serial Port Profile

The architecture of a Bluetooth Classic device is essentially the same as that of a BLE device. It is composed of the same four layers.

|  |  |  |
| --- | --- | --- |
|  | Application | The code that you write to implement your system functionality |
| Bluetooth Classic Stack | Host | Provides multiple connection paths with to the application each with its own properties (reliable, ordered, time critical etc). It also provides Services to the local and remote application |
| Controller | Establishes and maintain links between devices |
| Hardware | Radio | RF magic & the best reason to use Cypress chips |

Here is the overall picture of the simplest Bluetooth Classic system.



## Inquiry

The purpose of the Inquiry process is for a Bluetooth Master to find all the Bluetooth Slaves that are within its radio range that might provide some interesting Service. This is exactly the opposite of BLE where a Peripheral advertises it availability and the BLE Central Scans for those packets.

A Bluetooth Classic Slave sits in state called Inquiry Scan i.e. a listening only state, until it hears a Bluetooth Master broadcast an inquiry request message. The Slave application is responsible for putting the Stack into the Inquiry Scan state using the correct Stack API.

Upon hearing an Inquiry request the Slave will broadcast an Extended Inquiry Response (EIR) packet that contains its Name, Bluetooth Address (BDADR) and list of Services). These responses are handled completely by the Controller part of the Stack i.e. your Application is not aware of these Inquiry requests happening.

You should be aware that because of the vagaries of the Bluetooth Radio frequency hopping scheme, these Inquires make take up to 10ish seconds.

## Page / Connect

The Paging process is used for a Bluetooth Master to connect to a Bluetooth Slave. The Master is the "paging” the Slave device (remember the old school [pagers](https://www.youtube.com/watch?v=l7Og1DuMu3k&list=RDl7Og1DuMu3k&t=18)?).

A Bluetooth Classic Slave sits in state called Page Scan i.e. a listening only state, until a Bluetooth Master initiates the connection process by sending a Page request. The Slave application is responsible for putting the Stack into the Page Scan state using the correct Stack API.

A Slave can be in both the Page Scan and Inquiry Scan mode at the same time. Meaning a Master can initiate a connection to a Slave without Inquiring if it already knows of the existence of the Slave from a previous connection.

## Pair & Bond

The whole Bluetooth communication system depends on having a shared symmetric encryption key called the Link Key. Bluetooth Classic uses a process called Secure Simple Pairing that exchanges enough information the Link Key to be created.

The Secure Simple Pairing process was designed to minimize the chances that the communication link could be compromised by an eavesdropper or by a man-in-the-middle. The process is the same as the BLE process minus the Numeric Comparison method.

As with BLE, Bonding is just saving the BDADR/Link Key into a non-volatile memory so that it can be reused to speed up the re-initiating of a connection

## Discover the Services using Service Discovery Protocol (SDP)

A simple conceptual model of Bluetooth Classic device is a Server that is running one or more Services that are attached to Ports, the same model that we use in IP Networking.

One question that arises from this idea is how do I figure out “What Services are available and what Port are they listening on?”. The answer to both questions is the Service Discovery Protocol.

The SDP has a database embedded in it that contains a list of Service and what Port they are running on. And the SDP Protocol allows the Bluetooth Master to query the SDP database.

## Exchange Data with the Serial Port Profile

Once the Service Discovery is complete, the Bluetooth Master knows the Port number that it can connect to use the Serial Port Profile (SPP). The SPP is just one of these Servers (from the last section) that acts like a serial port. You put bytes in one side and they come out the other.

The Bluetooth Master then opens a connection to the SPP Server running on the Bluetooth Slave. At which point you can commence the final step in your first basic project, actually exchanging data.

# Secure Simple Pairing

Secure Simple Pairing is the same Pairing technique that we used in the BLE. You use a PIN code which is

1. Either ‘000000’ or trivial if you have no I/O capability
2. Displayed on one side, then entered on the other
3. Transmitted out of band (e.g. NFC)

The PIN is then used to encrypt random numbers which are generated on both side of the connection.

Finally, the Pin + Random Numbers + some data about the device are combined into a Link Key which serves as a shared secret to identify and encrypt data between the devices.

# Service Discovery Protocol (SDP)

From the Bluetooth Core Spec – “The service discovery protocol (SDP) provides a means for Applications to discover which Services are available and to determine the characteristics of those available services.” The SDP sits on top of the L2CAP layer – and when communicating generates a bunch of L2CAP traffic.

The Bluetooth SIG specifies that SDP database format in Volume 3 Part B of the Bluetooth Core Spec. The database is composed of one or more Service Records each containing one or more Service Attributes. Each Service Attribute is a Key/Value pair. There are several Bluetooth Sig Specified Service Attributes, in addition you can create custom Attributes.

The protocol provides for the Client to Search for Services and Attributes, and request the values of the same.

# L2CAP, RFCOMM & the Serial Port Profile

The Bluetooth Classic system has a stack of software and hardware built into it. For the purposes of this simple Bluetooth Classic example, three blocks in the Host are relevant, L2CAP, RFCOMM and the Serial Port Profile.

You can see the three blocks in this simplified diagram of the Stack.



## L2CAP

L2CAP is an acronym that stands for Logical Link Control and Adaptation-layer Protocol. L2CAP has one main function in the system. Specifically, it serves as a data packet multiplexor that lets you have streamed multiple connections from the higher level going into one interlaced set of packets going out the Radio. It obviously implements the de-multiplexor function as well, taking a single stream of packets and turning it back into complete streams on the other side of the link.

The L2CAP divides up the streams of data into L2CAP Channels that:

1. Divides up streams of data into smaller packets that will fit through the Radio
2. Provides quality of service to each of the L2CAP channels
3. Provide flow control

## RFCOMM

RFCOMM was built as a wired RS232 replacement protocol. The protocol support all the normal wires for a serial port include Rx, Tx, CTS, RTS, DSR, DTR, CD and Ri. Depending on the implementation, RFCOMM gives you up to 60 Server Channels of streams of serial data. The protocol is built on top of the L2CAP (a packet based system). It appears to the Application developer with APIs that look like a UART.

## Serial Port Profile

The Serial Port Profile specifies all the steps required to setup, discover and connect two virtual serial port over an RFCOMM connection.

# WICED Bluetooth Designer

# WICED Classic Bluetooth Firmware Architecture

# Exercises

Run the sample spp   
Attach with pc  
Attach with android   
Add a keyboard based pin code  
Add display for pin code  
Add support for multiple Pairing/Bonding