Bluetooth based location

RSSI measurements for different distances and antenna orientation and type

presented by
Tobias Karl
Mats Sandberg
Elizabeth Tse

Introduction:

The question was put forth, “was it possible to use Bluetooth as a location device”. In the past it has been possible to use Bluetooth as location device, but only to see if a device is in a range for a receiver, not exactly where the device was in that range. Our task was to see if the location of a device could be determined within a room or area, using the command RSSI.

Theory:

Bluetooth is a short-range radio frequency (RF) technology that operates at 2.4 GHz that is capable of transmitting voice and data. The effective range of Bluetooth devices is 32 feet (10 meters).

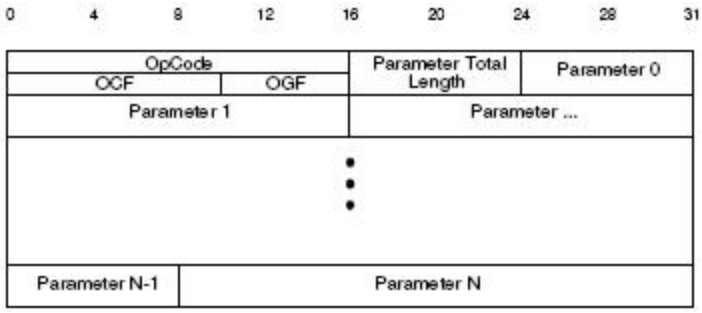
Host Control Interface (HCI)

Provides a uniform interface of accessing Bluetooth hardware via the Link-Manager, Baseband controller and hardware status, by using serial interfaces like USB, RS232 or PC-Card to control registers of the Bluetooth device. This communication with the HCI is entirely packet based, accomplished by using different types of packets to perform command requests and responses events.

Generating a HCI-Commands and analyzing a HCI-Event packets

Generating a HCI Command:

Command packets consist of 3 bytes header information and several bytes of command parameters. The pack begins with a 2 byte operations code (OpCode), split into a 10 bit operation command field (OCF) and a 6 bit operation group code (OGC). This is followed by a Parameter Total Length code which indicates the number of bytes used for command parameters.



The OpCode is generated by a process of taking the OCF and OGF and adding them together. This is done by taking the hex number for the OCF, turning it into a binary (big endian), take this number and add 2 zeros to the left hand side, and then turn this number into binary (little endian). The OGF is turned from a hex number to a decimal (big endian), the left most 2 digits are removed, this number is then changed into a decimal (little endian).

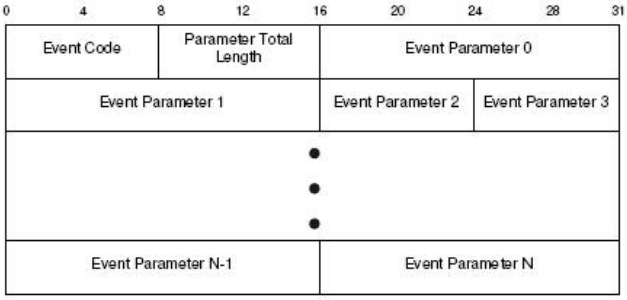
The OpCode is now computed by adding these to values as two 8 bit numbers.

OCF: XXXXXXXXXX
OGF: YYYYYY

OpCode: XXXXXXXX XXYYYYYY

Since both these numbers are binary little endian, they must be changed to binary big endian, and from this binary (big endian) number, the HEX values can then be obtained.

Analyzing a HCI-Event packet:
The first three parameters of the event packet consist of several parameters describing the command packet which caused this event packet. The last parameters of the event code are the return parameters requested by the command packet.



All of the HEX values for all the parameters are given in the Bluetooth Specification documents.

Commands used:

Read_Local_Name:
The Read_Local_Name command reads the stored user friendly name of the Bluetooth device. This user friendly name allows the user the to distinguish one Bluetooth device from another. The Name return parameter is a UTF-8 encoded string with a length of up to 248 bytes. The Name return parameter uses a null terminated (0x00) if the string is less than 248 bytes.

The Name Parameter is a string parameter and thus the Endianess does not apply to the Name Parameter. The first byte of the name is received first.

The OGF is set to 0x03

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Local_Name	0x0014		Status, Name

Return Parameters:
Status: *Size: 1 Byte*

Value	Parameter Description
0x00	Read_Local_Name command succeeded
0x01-0xFF	Read_Local_Name command failed see Table 6.1 on page 767 for list of Error Codes

Name: *Size: 248 Bytes*

Value	Parameter Description
	A UTF-8 encoded User Friendly Descriptive Name for the device. If the name contained in the parameter is shorter than 248 bytes, the end of the name is indicated by a NULL byte (0x00), and the following bytes (to fill up 248 bytes, which is the length of the parameter) do not have valid values.

Read_BD_ADDR:
The Read_BD_ADDR reads the value for the BD_ADDR parameter. BD_ADDR is 48-bit unique identifier for the unique Bluetooth device.

OGF is defined as 0x04

Command	OCF	Command Parameters	Return Parameters
HCI_Read_BD_ADDR	0x0009		Status, BD_ADDR

There are no Command Parameters.

Return Parameters:
Status: *Size: 1 Byte*

Value	Parameter Description
0x00	Read_BD_ADDR command succeeded.
0x01-0xFF	Read_BD_ADDR command failed. See Table 6.1 on page 766 for list of Error Codes.

BD_ADDR: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device

Write_Scan_Enable:
The Write_Scan_Enable command writes the value for the Scan_Enable parameter. This parameter controls whether or not the Bluetooth device will periodically scan for page attempts and/or inquiry requests from other Bluetooth devices. When the Page_Scan is enabled, the device enters a page scan mode based on the value of the Page_Scan_interval and Page_Scan_Window parameters. When the Inquiry_Scan is enabled, the device enters Inquiry Scan mode based on the value of the Inquiry_Scan_Interval and the Inquiry_Scan_Window parameters.

OGF is defined as 0x03

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Scan_Enable	0x001A	Scan_Enable	Status

Command parameter: Scan_Enable

Scan_Enable: *Size: 1 Byte*

Value	Parameter Description
0x00	No Scans enabled. **Default.**
0x01	Inquiry Scan enabled. Page Scan disabled.
0x02	Inquiry Scan disabled. Page Scan enabled.
0x03	Inquiry Scan enabled. Page Scan enabled.
0x04-0xFF	Reserved

Return parameter: Status

Status: *Size: 1 Byte*

Value	Parameter Description
0x00	Write_Scan_Enable command succeeded.
0x01-0xFF	Write_Scan_Enable command failed. See Table 6.1 on page 766 for list of Error Codes.

HCI_Inquiry:

The HCI_Inquiry command allows the Controller to control connection to another Bluetooth device. The Link Manager (LM) controls how the Bluetooth piconets and scatternet are established and maintained. These commands instruct the LM how to create and modify connection at the link layer with the Bluetooth remote device, to perform Inquiries to find other Bluetooth devices in range and other LMP commands.

The OGF is set to 0x01

Command	OCF	Command Parameters	Return Parameters
HCI_Inquiry	0x0001	LAP, Inquiry_Length, Num_Responses	

This command causes the Bluetooth device to enter Inquiry Mode where Bluetooth devices which are nearby maybe discovered.

Command Parameter LAP, Inquiry_Length, Num_Responses

The LAP input parameter contains the LAP from where the inquiry access code will be derived when the inquiry procedure is made.

LAP: *Size: 3 Bytes*

Value	Parameter Description
0x9E8B00– 0X9E8B3F	This is the LAP from which the inquiry access code should be derived when the inquiry procedure is made; see "Bluetooth Assigned Numbers" (http://www.bluetooth.org/assigned-numbers.htm).

The Inquiry_Length parameter specifies the total durations of the Inquiry Mode.

Inquiry_Length: *Size: 1 Byte*

Value	Parameter Description
N = 0xXX	Maximum amount of time specified before the Inquiry is halted. Size: 1 byte Range: 0x01 – 0x30 Time = N * 1.28 sec Range: 1.28 – 61.44 Sec

The Num_Response parameter specifies the number of responses that can be received before the Inquiry is ended.

Num_Responses: *Size: 1 Byte*

Value	Parameter Description
0x00	Unlimited number of responses.
0xXX	Maximum number of responses from the Inquiry before the Inquiry is halted. Range: 0x01 – 0xFF

A Command Status event is sent from the Controller to the host once the Inquiry command has been started by the blue tooth device. Once the Inquiry is complete the Controller will send a Inquiry complete event to the host, containing a summary of the results from the Inquiry process. The summary reports the number of nearby Bluetooth devices that have responded.

Remote_Name_Request:

The Remote_Name_Request command obtains the user friendly name of another Bluetooth devivce. The user friendly name is used to distinguish one Bluetooth device from another.

The OGF is set to 0x01

Command	OCF	Command Parameters	Return Parameters
HCI_Remote_Name_Request	0x0019	BD_ADDR, Page_Scan_Repetition_Mode, Page_Scan_Mode, Clock_Offset	

The BD_ADDR command parameter used to identify the device for

BD_ADDR: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for the device whose name is requested.

The Page_Scan_Repetition_Mode and Page_Scan_Mode command parameters specify the page scan modes supported by remote devices with the BD_ADDR. This BD_ADDR is acquired from the inquiry process.

Page_Scan_Repetition_Mode: *Size: 1 Byte*

Value	Parameter Description
0x00	R0
0x01	R1
0x02	R2
0x03 – 0xFF	Reserved.

Page_Scan_Mode: *Size: 1 Byte*

Value	Parameter Description
0x00	Mandatory Page Scan Mode.
0x01	Optional Page Scan Mode I.
0x02	Optional Page Scan Mode II.
0x03	Optional Page Scan Mode III.
0x04 – 0xFF	Reserved.

The Clock_Offset parameter value is the difference between the local clock and the clock of the remote device. The second through sixteenth bits of the difference are used. These are mapped to this parameter as bits 0 through 14 respectively. There is a Clock_Offset_Valid_Flag used to indicate if the Clock Offset is valid or not. This is located in bit 15 of the Clock_Offset command parameter.

Clock_Offset: *Size: 2 Bytes*

Bit format	Parameter Description
Bit 14.0	Bit 16.2 of CLKslave-CLKmaster.
Bit 15	Clock_Offset_Valid_Flag Invalid Clock Offfset = 0 Valid Clock Offset = 1

If there is no connection between the local device and the remote device, a temporary link layer connection will be established to obtain the name of the remote device.

The Host Controller sends the Command Status event to the Host when the Host Controller receives the Remote_Name_Request command. Link Manager once it has completed the LMP messages to retrieve the remote name, the local Bluetooth device's Host Controller will send a Remote Name Request Complete event to the Host.

No Command Complete event will be sent by the Host Controller, only the Remote Name Request Complete event will indicate that this command has been completed.

Create_Connection:
The Create_Connection command causes the Link Manager to create a connection to the Bluetooth device with the BD_ADDR used in the command parameters. The device then begins the Page process to create a link level connection. Link Manage determines via the current state of the device, its piconet and state of the remote device to be connected to, how the new ACL connection is to be established.

The OGF is set to 0x01

Command	OCF	Command Parameters	Return Parameters
HCI_Create_Connection	0x0005	BD_ADDR, Packet_Type, Page_Scan_Repetition_Mode, Page_Scan_Mode, Clock_Offset, Allow_Role_Switch	

Command parameters: BD_ADDR, Packet_Type, Page_Scan_Repetition_Mode, Page_Scan_Mode, Clock_Offest, Allow_Role_Switch

BD_ADDR: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device to be connected.

The Packet_Type command parameter specifies the only packet types the Link Manager can use for the ACL connection for sending HCI ACL Data Packets. Multiply packet types can be specified by using a bit-wise OR operation for the Packet Type parameter. The Link Manager has the opportunity to choose which packet type to be used from the list of acceptable packet types.

Packet_Type: *Size: 2 Bytes*

Value	Parameter Description
0x0001	Reserved for future use.
0x0002	Reserved for future use.
0x0004	Reserved for future use.
0x0008	DM1
0x0010	DH1
0x0020	Reserved for future use.
0x0040	Reserved for future use.
0x0080	Reserved for future use.
0x0100	Reserved for future use.
0x0200	Reserved for future use.
0x0400	DM3
0x0800	DH3
0x1000	Reserved for future use.
0x2000	Reserved for future use.
0x4000	DM5
0x8000	DH5

The Page_Scan_Repetition_Mode and Page_Scan_Mode parameters specify the page scan modes allowed by the remote device. This information is obtained during the inquiry process.

Page_Scan_Repetition_Mode: *Size: 1 Byte*

Value	Parameter Description
0x00	R0
0x01	R1
0x02	R2
0x03 – 0xFF	Reserved.

Page_Scan_Mode: *Size: 1 Byte*

Value	Parameter Description
0x00	Mandatory Page Scan Mode.
0x01	Optional Page Scan Mode I.
0x02	Optional Page Scan Mode II.
0x03	Optional Page Scan Mode III.
0x04 – 0xFF	Reserved.

The Clock_Offset parameter is the difference between its clock and the clock of the remote device. Only the second bit through sixteenth of the difference is used. These values are mapped to this parameter as bits 0 through 14 respectively. A Clock_Offset_Valid_Flag indicating if a Clock Offset is valid or not is located in bit 15 of the Clock_Offset parameter. A Connection handle for this connection is returned in the Connection Complete event.

Clock_Offset: *Size: 2 Bytes*

Bit format	Parameter Description
Bit 14.0	Bit 16.2 of CLKslave-CLKmaster.
Bit 15	Clock_Offset_Valid_Flag Invalid Clock Offfset = 0 Valid Clock Offset = 1

The Allow_Role_Switch parameter specifies if the request of a master-slave role switch when the remote device requests it at the connection setup, is accepts or rejects by the local device. This is requested by the remote device in the Role parameter of the Accept_Connection_Request command.

Allow_Role_Switch: *Size: 1 Byte*

Value	Parameter Description
0x00	The local device will be a master, and will not accept a master-slave switch requested by the remote device at the connection setup.
0x01	The local device may be a master, or may become a slave after accepting a master-slave switch requested by the remote device at the connection setup.
0x02-0xFF	Reserved for future use.

At least one packet type must be specified.
The Host should enable as many packet types the local device supports as possible for the Link Manager to perform efficiently.

Return parameters: There are none

When the Host Controller sends the Command Status event to the Host, the Host Controller receives the Create Connection command. Once the LM determines the connection was established, both Bluetooth devices Host Controller will send a Connection Complete event containing the Connection Handle to each Host.

Accept_Connection_Requeset:
The Accept_Connection_Request command returns the BD_ADDR of the device who sent a Connection Request, if the requested is excepted. This causes the Link Manage determine with regards to the current state of the device, its piconet and the state of the device to connect with, how to create a connection by the .

The OGF is set to 0x01

Command	OCF	Command Parameters	Return Parameters
HCI_Accept_Connection Request	0x0009	BD_ADDR, Role	

Command parameters: BD_ADDR, Role

BD_ADDR: *Size: 6 Bytes*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device to be connected

The Role command parameter indicates if the Link Manager shall perform a Master-Slave switch, and become the Master for this connection. The decision to accept a connection must be completed before the connection accept timeout expires on the local Bluetooth Module.

Read_RSSI:
The Read_RSSI command reads the value of the difference between the measure Received Signal Strength Indication and the limits of the Golden Receive Power Range for a connection handle to another Bluetooth device. The connection handle which can be found with the inquiry command is needed to identify the connect to which device you wish to find the value for. This connection Handle must be for an ACL connection.

A positive value returned by the RSSI command indicates how many dB the RSSI is above the upper limit for the Golden Receiver Power Range. A negative value returned by the RSSI command indicated how many dB the RSSI is below the Golden Receiver Range. If a zero value is returned, the RSSI is inside this Golden Receiver Power Range.

The accuracy of the dB reading is dependent on the Bluetooth hardware.

The OGF is set to 0x05

Command	OCF	Command Parameters	Return Parameters
HCI_Read_RSSI	0x0005	Connection_Handle	Status, Connection_Handle,RSSI

The return parameters Connection_Handle

Connection_Handle: *Size: 2 Bytes (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection Handle for the Connection for which the RSSI is to be read. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

The return parameters: Status, Connection_Handle and RSSI

Status: *Size: 1 Byte*

Value	Parameter Description
0x00	Read_RSSI command succeeded.
0x01-0xFF	Read_RSSI command failed. See Table 6.1 on page 766 for list of Error Codes.

Connection_Handle: *Size: 2 Bytes (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection Handle for the Connection for which the RSSI has been read. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

RSSI: *Size: 1 Byte*

Value	Parameter Description
N = 0xXX	Size: 1 Byte (signed integer) Range: -128 ≤ N ≤ 127 Units: dB

Read_Transmit_Power_level:

The Read_Transmit_Power_Level reads the value for the Transmit_Power_Level parameter for a specified Connection Handle. This Connection handle must be an ACL connection.

OGF is defined as 0x03

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Transmit_Power_Level	0x002D	Connection_Handle, Type	Status, Connection_Handle, Transmit_Power_Level

Command Parameters: Connection_Handle and Type

Connection_Handle: *Size: 2 Bytes (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection Handle's Transmit Power Level setting to read. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Type: *Size: 1 Byte*

Value	Parameter Description
0x00	Read Current Transmit Power Level.
0x01	Read Maximum Transmit Power Level.
0x02-0xFF	Reserved

Return parameters: Status, Connection_Handle and Tranmit_Power_Level

Status: *Size: 1 Byte*

Value	Parameter Description
0x00	Read_Transmit_Power_Level command succeeded.
0x01-0xFF	Read_Transmit_Power_Level command failed. See Table 6.1 on page 766 for list of Error Codes.

Connection_Handle: *Size: 2 Bytes (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection Handle's Transmit Power Level setting is returned. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Transmit_Power_Level: *Size: 1 Byte*

Value	Parameter Description
N = 0xXX	Size: 1 Byte (signed integer) Range: -30 ≤ N ≤ 20 Units: dBm

A Command Complete event is generated when the Read_Transmit_Power_Level command has completed.

Get_Link_Quality:

The Get_Link_Quality returns a value for the Link_Quality for a specified Connection Handle. This command returns a value between 0-255, representing the quality of the link between two Bluetooth devices. The better the quality the higher the value will be. This value measurement is set by the vendor.

The OGF is set to 0x05

Command	OCF	Command Parameters	Return Parameters
HCI_Get_Link_Quality	0x0003	Connection_Handle	Status, Connection_Handle, Link_Quality

Command Parameters:

Connection_Handle: *Size: 2 Bytes (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection Handle for the connection for which link quality parameters are to be read. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters: Status, Connection_Handle and Link_Quality

Status: *Size: 1 Byte*

Value	Parameter Description
0x00	Get_Link_Quality command succeeded.
0x01-0xFF	Get_Link_Quality command failed.See Table 6.1 on page 766 for list of Error Codes.

Connection_Handle: *Size: 2 Bytes (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection Handle for the connection for which the link quality parameter has been read. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Link_Quality: *Size: 1 Byte*

Value	Parameter Description
0xXX	The current quality of the Link connection between the local device and the remote device specified by the Connection Handle Range: 0x00 – 0xFF The higher the value, the better the link quality is.

A Command Complete event is generated when the Get_Link_Quality command has completed.

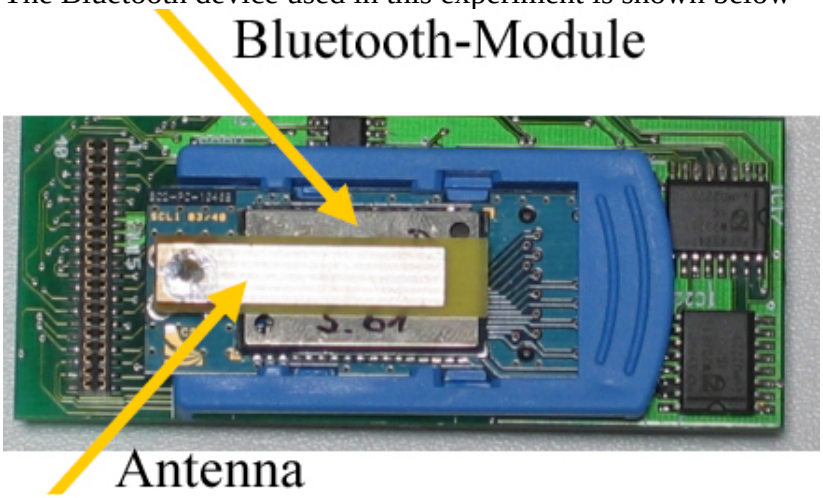
Method of finding the Bluetooth devices location:

With the data from the RSSI, Read_Power_Transmited_Level and Get_Link_Quality, it should be possible to find the distance apart the Bluetooth devices are from each other. If data is taken at measured distances that Bluetooth devices are apart, then the Read_RSSI command can be used to get the strength of the signal at each measured distance. Once these values are obtained, it is then possible to use these values as a guideline to judge how far away the Bluetooth devices are from each other. If four devices are used, then a triangular method of location can be use to not just determine the distance away from each Bluetooth device but also the relative location with regards to all the Bluetooth devices. This would allow three devices to be in a fixed know locations and the fourth be the unknown location device. With the RSSI values from connection to the three fixed Bluetooth device, it should be possible to find the actual location (within a measure of error) of where the fourth Bluetooth device is location.

Another way to find the locations, is to use the Read_Power_Transmited_Power and Get_Link_Quality, to find the distance. When there is a constant Read_Power_Transmited_Power value, the Get_Link_Quality can then be used to judge distance, the same way described above for the Read_RSSI command.

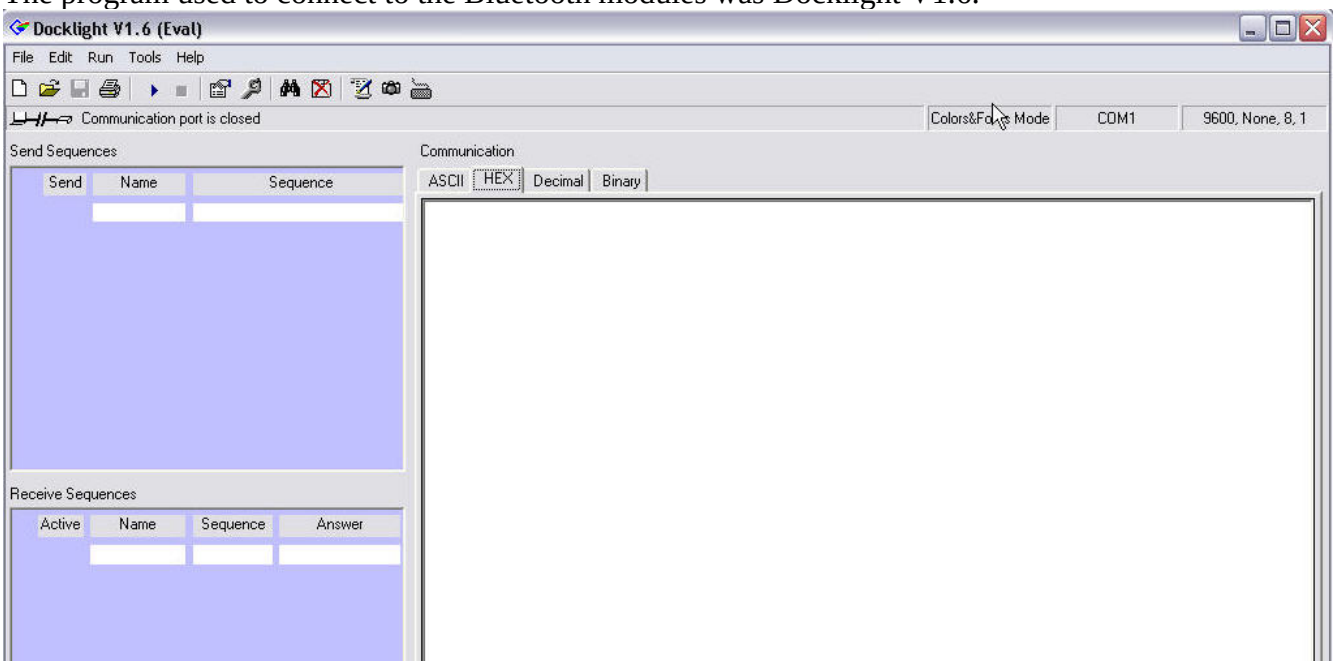
Procedure:

The Bluetooth device used in this experiment is shown below



The two Bluetooth devices must first connect to each other, one must send a packet to determine the Transmitted power level, the RSSI and the Quality of signal can all be obtained.

The program used to connect to the Bluetooth modules was Docklight V1.6.



The first step was to create a connection between the two devices. To accomplish this, some information was need. The name and address of both devices, both devices must enable Write_Scan_Enable, and an HCI_Inquiry command must be made to obtain the clock offset. Once this information is obtained then the Create_Connection command request can be made from one device to the other, and the receiving device must then send a Accept_Request command.

The steps involved in creating a connection:

The local name of both Bluetooth devices can be obtained using Read_Local_name on both devices.

OCF	=	0x0014	=	0000010100	Big endian
				0010100000	Little endian
OGF	=	0x03	=	000011	Big endian
				110000	Little endian
OpCode				00101000 00110000	Little endian
				00010100 00001100	Big endian
				14 0C	

The Read_Local_name: 01 14 0C 00

- 01 Command Event
- 14 0C OpCode
- 00 Number of parameters

The address of both Bluetooth devices can be obtain by using Read_Local_Address.

OCF	=	0x0009	=	0000001001	Big endian
				1001000000	Little endian
OGF	=	0x04	=	000100	Big endian
				001000	Little endian
OpCode				10010000 00001000	Little endian
				00001001 00010000	Big endian
				09 10	

OpCode is 09 10

The Read_Local_Address: 01 09 10 00

- 01 Command event
- 09 10 OpCode
- 00 Number of parameters

Write_Scan_Enable

OCF	=	0x001A	=	0000011010	Big endian
				0101100000	Little endian
OGF	=	0x03	=	000011	Big endian
				110000	Little endian
OpCode				01011000 00110000	Little endian
				00011010 00001100	Big endian
				1A 0C	

Write_Scan_Enable: 01 1A 0C 01 03

- 01 Command event
- 1A 0C OpCode
- 01 Number of Parameter
- 03 Parameters (enable Inquiry Scan and Page Scan)

To find other devices, the Inquiry command is used by both devices

OCF	=	0x0001	=	0000000001	Big endian
				1000000000	Little endian
OGF	=	0x01	=	000001	Big endian
				100000	Little endian
OpCode				10000000 00100000	Little endian
				00000001 00000100	Big endian
				01 04	

HCI_Inquiry: 01 01 04 04

01	Command event
01 04	OpCode
04	Number of devices to look for

The response from the HCI_Inquiry will give the clock offset and the

And example response

04 02 0F 01 B8 76 4D 9F 11 00 01 00 00 04 02 52 91 62

01	Number of Responses
B8 76 4D 9F 11 00	Bluetooth Device Address
04 02 52	Class of Device
91 62	Clock offset

the numbers needed here for the next command are the Bluetooth device address and the Clock offset.

To create a connection, the Create_Connection command is used by device

OCF	=	0x0005	=	0000000101	Big endian
				1010000000	Little endian
OGF	=	0x01	=	000001	Big endian
				100000	Little endian
OpCode				10100000 00100000	Little endian
				00000101 00000100	Big endian
				05 04	

Create_Connection: 01 05 04 0D XX XX XX XX XX XX 10 00 01 00 XX XX 00

01	
05 04	OpCode
0D	parameter length
XX XX XX XX XX XX	remote Bluetooth device
10 00	Packet type
XX XX	Clock offset

Example of a responses:

Status Event: 04 0F 04 00 01 05 04

Accepting a connection is done with the Accept_Connection_Request.

OCF	=	0x0009	=	0000001001	Big endian
				1001000000	Little endian
OGF	=	0x01	=	000001	Big endian
				100000	Little endian
OpCode				10010000 00100000	Little endian
				00001001 00000100	Big endian
				09 04	

Create_Connection: 01 09 04 07 XX XX XX XX XX XX 01

01	Command Event
09 04	OpCode
07	number of parameters
XX XX XX XX XX XX	Bluetooth Address
01	Slave or Master setting

Responses:

First response: 04 0F 04 00 01 09 04

Second response: 04 03 0B 00 XX XX XX XX XX XX XX 01 00

00	Connection OK
XX XX	Connection Handle
XX XX XX XX XX XX	Bluetooth Address
01	Connection Complete event

A connection is now established. The next thing to do is send a data packet and check the RSSI, Read_Transmit_Power_Level, and Get_Link_Quality.

To send a data packet a data packet with a simple message of HELLO

Connection Handle: 29 00

Packet boundary flag: 10

Broadcast flag: 00

00 29	00 10
0000 00101001	0010
10010100 0000	0100
10010100	00000100
00101001	00100000
29	20

02 29 20 05 00 48 45 4C 4C 4F

02	ACL
29 20	Connection Handle

05 00 Data Length
00 'broadcast flag' with value 00
48 45 4C 4C 4F message

To read the RSSI value, use the Read_RSSI command

OCF	=	0x0014	=	0000000101	Big endian	
				1010000000	Little endian	
OGF	=	0x05	=	000101	Big endian	
				101000	Little endian	
OpCode				10100000	00101000	Little endian
				00000101	00101000	Big endian
				05	14	

Read_RSSI: 01 05 14 02 29 00

01 Event Command
05 14 OpCode
02 Following bits
29 00 Connection Handle

The next value to check is the power level of the transmitted signal, this can be done with the command Read_Transmited_Power_Level.

Read_Transmited_Power_Level: 01 2D 0C 03 XX XX 00

01 Event Command
2D 0C OpCode
03 Number of parameters
XX XX Connection handle

Example Response:
04 0E 07 01 2D 0C 00 XX XX XX

00 Status OK
XX XX Connection Handle
XX Power Level

The last value we were interested in was the Get_Link_Quality

OCF	=	0x0003	=	0000000011	Big endian	
				1100000000	Little endian	
OGF	=	0x05	=	000101	Big endian	
				101000	Little endian	
OpCode				11000000	00101000	Little endian
				00000011	00010100	Big endian
				03	14	

Get_Link_Quality: 01 03 14 02 29 00

01	Event Command
03 14	OpCode
02	Number of parameters
29 00	Connection Handle

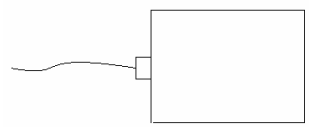
Results:

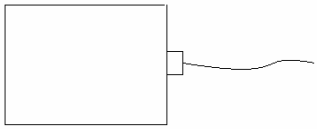
The name of our devices: Luke Skywalker and Johanna

The address of Luke Skywalker: 53 70 01 5B 02 00
The address of Johanna: 46 B2 19 C2 50 00

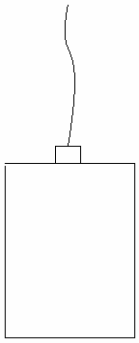
The clock offset: this value changed very time we reconnected the devices or when the devices were reset. We did have a problem with one device, the power connection to the device had a lose connection (we believe) and any movement of the device would cut off the power to the device for a short time and the device would reset. This resulted in the Write_Scan_enable command and the Inquiry command had to be done again, to obtain the new clock offset value which is needed for the Create_Connection command.

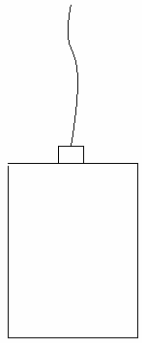
RSSI results:





Distance	0cm	5cm	10cm	15cm	20cm	25cm	30cm
RSSI (send)	18	12	08	08	06	05	00
RSSI (recieve)	1A	13	0F	0C	07	02	00





Distance	2.5	5cm	10cm	15cm	20cm
RSSI (send)	18	12	08	08	06
RSSI (recieve)	1A	13	0F	0C	07

Transmit Power Level:

Distance	0m	1m	2m	3m	6m
Power Level	E8	EC	F8	FC	???

Link Quality:

We were unable to get any measurement over 6m. The cords to our devices and the computers in the project room didn't allow for us to measure any distance over 6m.

Conclusion:

We were not able to find the location of a Bluetooth device in an area or room. Our inability to do this was due to a few problems.

The first problem, was the Golden Receiver Power Range. This range was too large. If the range had been set to a small range, we would have been able to get more results above and below this Golden Receiver Power Range. The Golden Receiver Power Range that was set in our device was too large, every value over 30cm and to the distance we could separate our devices, all gave 00 as a result. The 00 meant we were in the Golden Receiver Power Range, so any measurements using RSSI could only be made between 0 and 30cm.

The second problem we encountered was the length of our cords of the devices and the layout of the computer in the project room. No matter how we tried to arrange the Bluetooth devices we were unable to get a distance longer then 6m.

If these two problems could have been solved or change, we might have been able to use the Read_RSSI, Link Quality and Power Read_Transmit_Power_level and Get_Link_Quality to obtain a value to estimate where a device was in the Bluetooth receivers range.