Size *Type: uint* *Size: 2 octets*

Value	Description
0xXXXX	Size, in bytes, of the data in the buffer.

10.7.15 GetInfo

Table 104 shows the parameters associated with the L2CA GetInfo primitive.

Table 104—GetInfo

Service	Input parameters	Output parameters
L2CA_GetInfo	BD_ADDR, InfoType	Result, InfoData, Size

Description:

This primitive represents the initiation of an L2CA_InfoReq messageand the reception of the corresponding L2CAP_InfoRsp message.

Input Parameters:

BD_ADDR *Type: uint* *Size: 6 octets*

Value	Description
0xXXXXXXXXXXXX	Unique Bluetooth address of target device

InfoType *Type: uint* *Size: 2 octets*

Value	Description
0x0001	Maximum connectionless MTU size

Output Parameters:

Result *Type: uint* *Size: 2 octets*

Value	Description
0x0000	Response received
0x0001	Not supported
0x0002	Informational PDU rejected, not supported by remote device
0x0003	Timeout occurred

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 278 of 1169

InfoData *Type: pointer* *Size: N/A*

Value	Description
N/A	The buffer containing the contents received in the data payload of the Information Response command.

Size *Type: uint* *Size: 2 octets*

Value	Description
0xXXXX	Size, in bytes, of the data in the InfoData buffer.

10.7.16 Disable connectionless traffic

Table 105 shows the parameters associated with the L2CA disable connectionless traffic primitive.

Table 105—Disable connectionless traffic

Service	Input parameters	Output parameters
L2CA_DisableCLT	PSM	Result

Description:

General request to disable the reception of connectionless packets. The input parameter is the *PSM* value indicating service that should be blocked. This command may be used to incrementally disable a set of PSM values. The use of the "invalid" PSM 0x0000 blocks all connectionless traffic. The output parameter *Result* indicates the success or failure of the command. A limited device might support only general blocking rather than PSM-specific blocks and would fail to block a single nonzero PSM value.

Input Parameters:

PSM *Type: uint* *Size: 2 octets*

Value	Description
0x0000	Block all connectionless traffic
0xXXXX	Protocol/Service Multiplexer field to be blocked

Output Parameters:

Result *Type: uint* *Size: 2 octets*

Value	Description
0x0000	Successful
0x0001	Failure – not supported

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 279 of 1169

10.7.17 Enable connectionless traffic

Table 106 shows the parameters associated with the L2CA enable connectionless traffic primitive.

Table 106—Enable connectionless traffic

Service	Input parameters	Output parameters
L2CA_EnableCLT	PSM	Result

Description:

General request to enable the reception of connectionless packets. The input parameter is the *PSM* value indicating the service that should be unblocked. This command may be used to incrementally enable a set of PSM values. The use of the 'invalid' PSM 0x0000 enables all connectionless traffic. The output parameter *Result* indicates the success or failure of the command. A limited device might support only general enabling rather than PSM-specific filters, and would fail to enable a single non-zero PSM value.

Input Parameters:

PSM *Type: uint* *Size: 2 octets*

Value	Description
0x0000	Enable all connectionless traffic
0xXXXX	Protocol/Service Multiplexer field to enable

Output Parameters:

Result *Type: uint* *Size: 2 octets*

Value	Description
0x0000	Successful
0x0001	Failure – not supported

10.8 Summary

L2CAP is one of two link level protocols running over the Baseband. L2CAP is responsible for higher level protocol multiplexing, MTU abstraction, group management, and conveying quality of service information to the link level.

Protocol multiplexing is supported by defining channels. Each channel is bound to a single protocol in a many-to-one fashion. Multiple channels can be bound to the same protocol, but a channel cannot be bound to multiple protocols. Each L2CAP packet received on a channel is directed to the appropriate higher level protocol.

L2CAP abstracts the variable-sized packets used by the Baseband Protocol (see Clause 8). It supports large packet sizes up to 64 kilobytes using a low-overhead segmentation-and-reassembly mechanism.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 280 of 1169

Group management provides the abstraction of a group of units allowing more efficient mapping between groups and members of the Bluetooth piconet. Group communication is connectionless and unreliable. When composed of only a pair of units, groups provide connectionless channel alternative to L2CAP's connection-oriented channel.

L2CAP conveys QoS information across channels and provides some admission control to prevent additional channels from violating existing QoS contracts.

10.8.1 Example configuration MSCs

Figure 130 illustrates the basic configuration process. In this example, the devices exchange MTU information. All other values are assumed to be default.

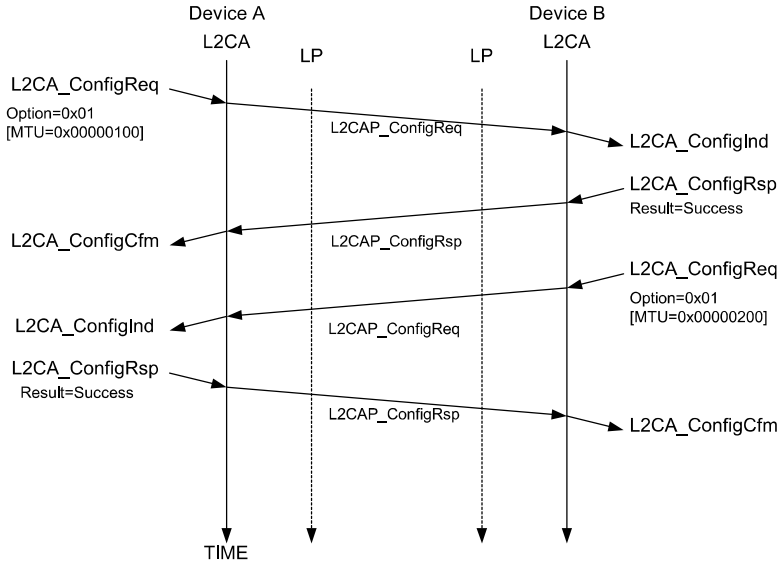


Figure 130—Basic MTU exchange

Figure 131 illustrates how two devices interoperate even though one device supports more options than the other does. Device A is an upgraded version. It uses a hypothetically defined option type 0x20 for link-level security. Device B rejects the command using the Configuration Response packet with result "unknown parameter" informing Device A that option 0x20 is not understood. Device A then resends the request omitting option 0x20. Device B notices that it does not need to such a large MTU and accepts the request but includes in the response the MTU option informing Device A that Device B will not send an L2CAP packet with a payload larger than 0x80 octets over this channel. On receipt of the response, Device A could reduce the buffer allocated to hold incoming traffic.

Copyright © 2002 IEEE. All rights reserved.

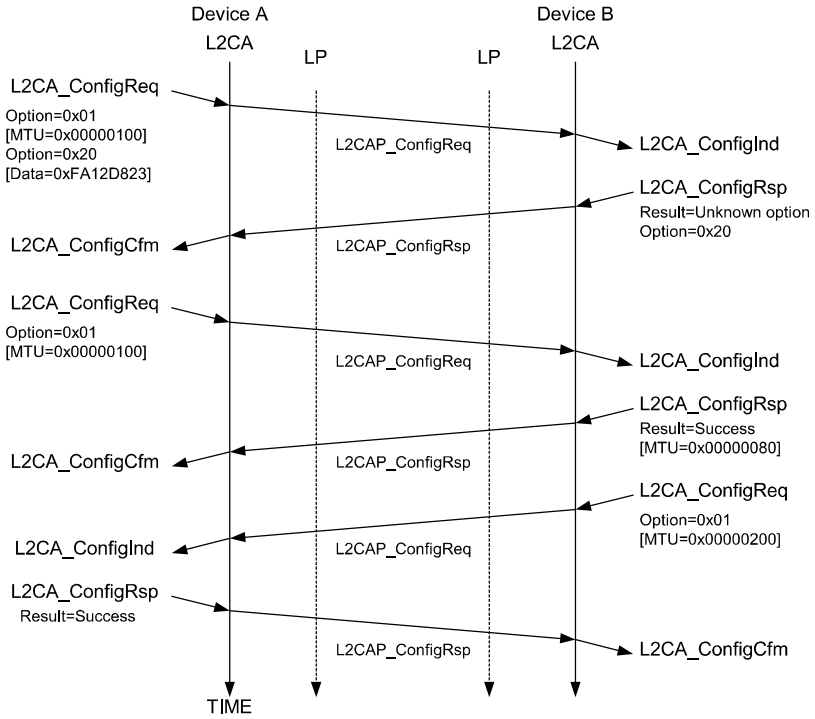


Figure 131—Dealing with unknown options

Figure 132 illustrates an unsuccessful configuration request. There are two problems described by this example. The first problem is that the configuration request is placed in an L2CAP packet that cannot be accepted by the remote device, due to its size. The remote device informs the sender of this problem using the Command Reject message. Device A then resends the configuration options using two smaller L2CAP_ConfigReq messages.

The second problem is an attempt to configure a channel with an invalid CID. For example, device B may not have an open connection on that CID (0x01234567 in this example case).

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 282 of 1169

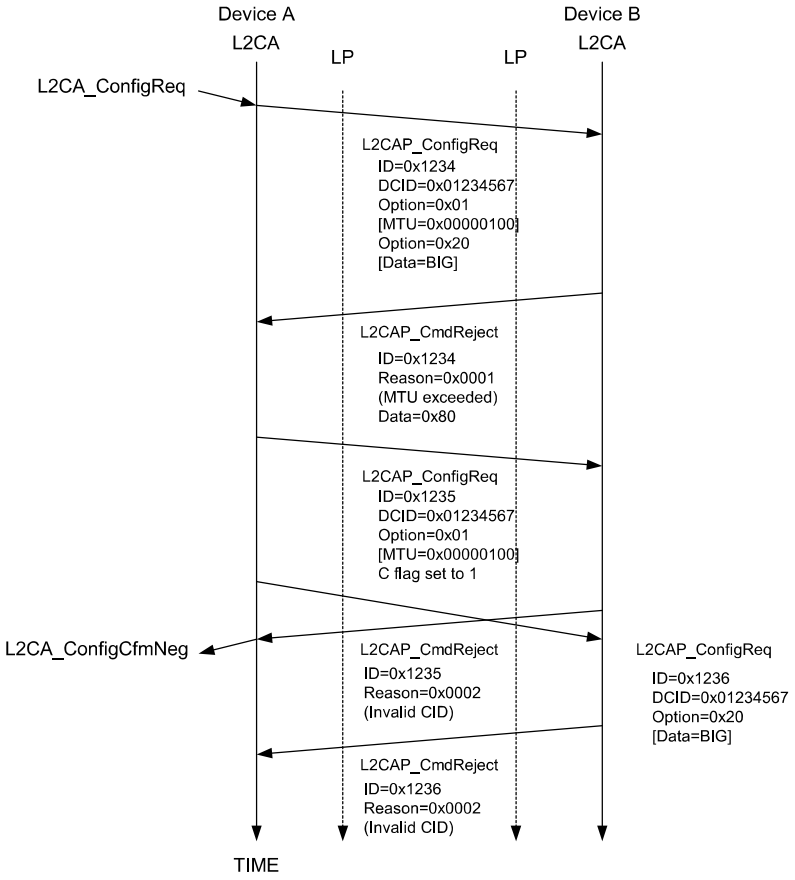


Figure 132—Unsuccessful configuration request

10.8.2 Implementation guidelines

This subclause contains some guidelines for implementations. These guidelines are not part of the compliance tests. At the moment they are simply suggestions on how to solve some difficult problems.

10.8.3 RTX timer

Implementations should not start this timer on an L2CAP Connection Request packet unless the physical link has been established. Otherwise the Baseband paging mechanism might increase the cost of the request beyond that of the minimal timeout value. If an implementation performs some form of security check it is recommended that the connection pending response be sent back prior to any consultation with a security manager that might perform Baseband authentication commands. If any security check requires user interaction, the link might timeout waiting for the user to enter a PIN.

10.8.4 QoS mapping to LM and L2CAP implementations

10.8.4.1 Token rate

The LM should ensure data is removed from the transmission buffer at this rate. The LM should ensure the polling interval is fast enough to support this data rate. The polling interval should be adjusted if the packet

Copyright © 2002 IEEE. All rights reserved.

type changes. If the buffer overflows, and the service type is Guaranteed, a QoS violation should be reported. If the service type is Best Effort, and a Token Rate was non-zero, a QoS violation should also be reported.

Given a Token Rate of 0xFFFFFFFF, and Service Type of Guaranteed, the LM should refuse any additional connections from remote devices and disable all periodic scans.

10.8.4.2 Token bucket size

L2CAP implementations should ensure that a buffer meeting the size request is allocated for the channel. If no buffer is available, and the service type is Guaranteed, the request should be rejected. If no appropriately sized buffer is available, and the service type is Best Effort, the largest available buffer should be allocated.

10.8.4.3 Peak bandwidth

If the token bucket buffer overflows, a QoS violation should be raised.

10.8.4.4 Latency

The LM should ensure the polling interval is at least this value. If the polling interval necessary to support the token rate is less than this value, the smaller interval should be used. If this interval cannot be supported, a QoS violation should be raised.

10.8.4.5 Delay variation

The LM may ignore this value because there is no clear mapping between L2CAP packet delays and the necessary polling interval without requiring the LM to comprehend the length field in L2CAP packets.

10.8.5 Collision tables

Table I: Result of Second Link Timeout Request

Current value	Requested value	Result
X	X	X
X	Y	If (X < Y) then X, else Y

Table II: Result of Second Flush Timeout Request

Current value	Requested value	Result
N	0	N
N	N	N
N	M ≠ N	Reject

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 284 of 1169

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 285 of 1169

11. Control interface

Figure 133 indicates the relationship of the Bluetooth protocol stack to this clause. This clause describes the functional specification for the Control interface for IEEE Std 802.15.1-2002. It is based upon the HCI section of the Bluetooth Specification. With the exception of 11.1, the text and graphics are solely an extraction from the original document. Normally this function would be refered to as a Management function in the IEEE nomenclature. The term Control interface was used to make clear the origin of the material.

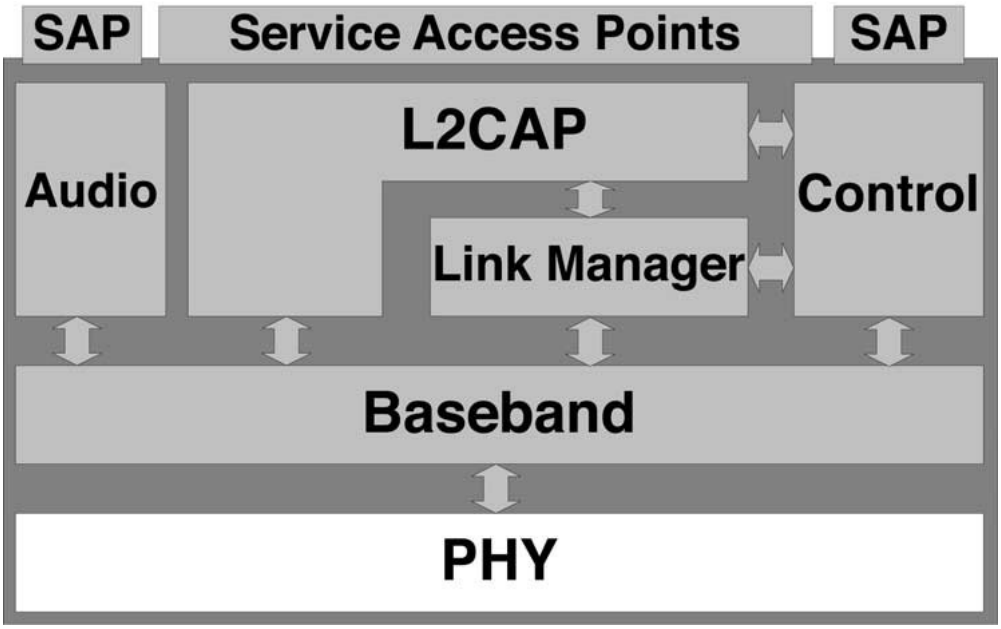


Figure 133—Control interface relationships

11.1 IEEE introduction

The HCI provides a command interface to the baseband controller and link manager, and access to hardware status and control registers. This interface provides a uniform method of accessing the Bluetooth baseband capabilities. The HCI section has two functions in the Bluetooth Specification:

1) Define a basis for a physical interface for a Bluetooth external module
2) Define the control functions necessary for all Bluetooth implementations

Since IEEE 802 Standards describe protocols and not implementations, the physical interface portions of the original document were omitted. Those portions that describe the logical relationship between the lower software layers and the controlling entity above it have been preserved.

The Host receives asynchronous notifications of HCI (management) events independent of which host controller transport layer is used. HCI events are used for notifying the Host when something occurs. When the Host discovers that an event has occurred, it will then parse the received event packet to determine which event occurred.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 286 of 1169

11.1.1 Differences between IEEE Std 802.15.1-2002 and the Bluetooth Specification

Table 107—Bluetooth commands omitted from IEEE Std 802.15.1-2002

Command	Meaning
Set_Host_Controller_To_Host_Flow_Control	The Set_Host_Controller_To_Host_Flow_Control command is used by the Host to turn flow control on or off in the direction from the Host Controller to the Host.
Host_Buffer_Size	The Host_Buffer_Size command is used by the Host to notify the Host Controller about its buffer sizes for ACL and SCO data. The Host Controller will segment the data to be transmitted from the Host Controller to the Host, so that data contained in HCI Data Packets will not exceed these sizes.
Host_Number_Of_Completed_Packets	The Host_Number_Of_Completed_Packets command is used by the Host to indicate to the Host Controller when the Host is ready to receive more HCI packets for any connection handle.
Read_SCO_Flow_Control_Enable	The Read_SCO_Flow_Control_Enable command provides the ability to read the SCO_Flow_Control_Enable setting. By using this setting, the Host can decide if the Host Controller will send Number Of Completed Packets events for SCO Connection Handles.
Write_SCO_Flow_Control_Enable	The Write_SCO_Flow_Control_Enable command provides the ability to write the SCO_Flow_Control_Enable setting. By using this setting, the Host can decide if the Host Controller will send Number Of Completed Packets events for SCO Connection Handles.

NOTE—The commands in Table 107 have been omitted from this standard, since they represent the physical interface implementation.

11.2 HCI commands

11.2.1 Introduction

The HCI provides a uniform command method of accessing the Bluetooth hardware capabilities. The HCI Link commands provide the Host with the ability to control the link layer connections to other Bluetooth devices. These commands typically involve the LM to exchange LMP commands with remote Bluetooth devices. For details, see Clause 9.

Copyright © 2002 IEEE. All rights reserved.

The HCI Policy commands are used to affect the behavior of the local and remote LM. These Policy commands provide the Host with methods of influencing how the LM manages the piconet. The Host Controller and Baseband, Informational, and Status commands provide the Host access to various registers in the Host Controller.

HCI commands may take different amounts of time to be completed. Therefore, the results of commands will be reported back to the Host in the form of an event. For example, for most HCI commands the Host Controller will generate the Command Complete event when a command is completed. This event contains the return parameters for the completed HCI command. For enabling the Host to detect errors on the HCI-Transport Layer, there needs to be a timeout between the transmission of the Host's command and the reception of the Host Controller's response (e.g., a Command Complete or Command Status event). Since the maximum response timeout is strongly dependent on the HCI-Transport Layer used, it is recommended to use a default value of 1 s for this timer. This amount of time is also dependent on the number of commands unprocessed in the command queue.

11.2.2 Terminology

Baseband packet: The smallest unit of data that is transmitted by one device to another, as defined in Clause 8.

Packet: A higher-level protocol message than the baseband packet, currently only L2CAP (see Clause 10) is defined, but additional packet types may be defined later.

Connection handle: A connection handle is a 12-bit identifier which is used to uniquely address a data/voice connection from one Bluetooth device to another. The connection handles can be visualized as identifying a unique data pipe that connects two Bluetooth devices. The connection handle is maintained for the lifetime of a connection, including when a device enters Park, Sniff, or Hold mode. The Connection Handle value has local scope between Host and Host Controller. There can be multiple connection handles for any given pair of Bluetooth devices but only one ACL connection.

Event: A mechanism that the HCI uses to notify the Host for command completion, link layer status changes, etc.

11.2.3 Data and parameter formats

- All values are in Binary and Hexadecimal Little Endian formats unless otherwise noted.
- In addition, all parameters which can have negative values shall use 2's complement when specifying values
- Arrayed parameters are specified using the following notation: ParameterA[i]. If more than one set of arrayed parameters are specified (e.g,. ParameterA[i], ParameterB[i]), then the order of the parameters are as follows: ParameterA[0], ParameterB[0], ParameterA[1], ParameterB[1], ParameterA[2], ParameterB[2], … ParameterA[n], ParameterB[n].
- Unless noted otherwise, all parameter values are sent and received in Little Endian format [i.e., for multibyte parameters the rightmost (least significant byte) is transmitted first].
- All command and event parameters that are not arrayed and all elements in an arrayed parameter have fixed sizes (an integer number of bytes). The parameters and the size of each not arrayed parameter (or of each element in an arrayed parameter) contained in a command or an event is specified for each command or event. The number of elements in an arrayed parameter is not fixed.
- Where bit strings are specified, the low order bit is the right hand bit, e.g., 0 is the low order bit in "10".

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 288 of 1169

11.2.4 Exchange of HCI-specific information

The Host Controller Transport Layer provides transparent exchange of HCI-specific information. These transporting mechanisms provide the ability for the Host to send HCI commands, ACL data, and SCO data to the Host Controller. These transport mechanisms also provide the ability for the Host to receive HCI events, ACL data, and SCO data from the Host Controller.

Since the Host Controller Transport Layer provides transparent exchange of HCI-specific information, the HCI specification specifies the format of the commands, events, and data exchange between the Host and the Host Controller. The next sections specify the HCI packet formats.

11.2.4.1 HCI command packet

The HCI Command Packet is used to send commands to the Host Controller from the Host. The format of the HCI Command Packet is shown in Figure 134, and the definition of each field is explained below. When the Host Controller completes most of the commands, a Command Complete event is sent to the Host. Some commands do not receive a Command Complete event when they have been completed. Instead, when the Host Controller receives one of these commands the Host Controller sends a Command Status event back to the Host when it has begun to execute the command. Later on, when the actions associated with the command have finished, an event that is associated with the sent command will be sent by the Host Controller to the Host. However, if the command does not begin to execute (there may be a parameter error or the command may currently not be allowed), the event associated with the sent command will not be returned. The Command Status event will, in this case, return the appropriate error code in the Status parameter. On initial power-on, and after a reset, the Host can send a maximum of one outstanding HCI Command Packet until a Command Complete or Command Status event has been received. If an error occurs for a command for which a Command Complete event is returned, the Return_Parameters field may not contain all the return parameters specified for the command. The Status parameter, which explains the error reason and which is the first return parameter, will always be returned. If there is a Connection_Handle parameter or a BD_ADDR parameter right after the Status parameter, this parameter will also be returned so that the Host can identify to which instance of a command the Command Complete event belongs. In this case, the Connection_Handle or BD_ADDR parameter will have exactly the same value as that in the corresponding command parameter. It is implementation specific whether more parameters will be returned in case of an error.

Note that the BD_ADDR return parameter of the command Read_BD_ADDR is not used to identify to which instance of the Read_BD_ADDR command the Command Complete event belongs. It is therefore not mandatory for the Host Controller to return this parameter in case of an error.

If an error occurs for a command for which no Command Complete event is returned, all parameters returned with the event associated with this command may not be valid. The Host shall take care as to which parameters may have valid values depending on the value of the Status parameter of the Complete event associated with the given command. The Command Complete and Command Status events contain a parameter called Num_HCI_Command_Packets, which indicates the number of HCI Command Packets the Host is currently allowed to send to the Host Controller. The Host Controller may buffer one or more HCI command packets, but the Host Controller shall start performing the commands in the order in which they are received. The Host Controller can start performing a command before it completes previous commands. Therefore, the commands do not always complete in the order they are started. The Host Controller shall be able to accept HCI Command Packets with up to 255 bytes of data excluding the HCI Command Packet header.

Each command is assigned a 2 byte Opcode used to uniquely identify different types of commands. The Opcode parameter is divided into two fields, called the OpCode Group Field (OGF) and OpCode Command Field (OCF). The OGF occupies the upper 6 bits of the Opcode, while the OCF occupies the remaining 10 bits. The OGF of 0x3F is reserved for vendor-specific debug commands. The OGF of 0x3E is reserved for

Copyright © 2002 IEEE. All rights reserved.

Bluetooth Logo Testing. The organization of the Opcodes allows additional information to be inferred without fully decoding the entire Opcode.

Note that the OGF composed of all "ones" has been reserved for vendor-specific debug commands. These commands are vendor-specific and are used during manufacturing, for a possible method for updating firmware, and for debugging.

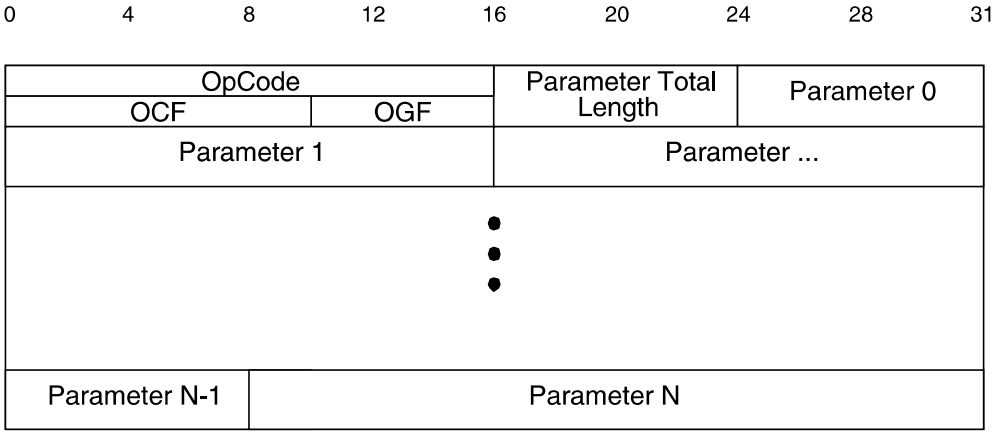


Figure 134—HCI command packet

Op_Code: *Size: 2 Bytes*

Value	Parameter description
0xXXXX	OGFRange (6 bits): 0x00–0x3F (0x3E reserved for Bluetooth logo testing and 0x3F reserved for vendor-specific debug commands). OCF Range (10 bits): 0x0000–0x03FF.

Parameter_Total_Length: *Size: 1 Byte*

Value	Parameter description
0xXX	Lengths of all of the parameters contained in this packet measured in bytes. (N.B.: total length of parameters, not number of parameters).

Parameter 0–N: *Size: Parameter Total Length*

Value	Parameter description
0xXX	Each command has a specific number of parameters associated with it. These parameters and the size of each of the parameters are defined for each command. Each parameter is an integer number of bytes in size.

11.2.4.2 HCI event packet

The HCI Event Packet is used by the Host Controller to notify the Host when events occur. The Host shall be able to accept HCI Event Packets with up to 255 bytes of data excluding the HCI Event Packet header. The format of the HCI Event Packet is shown in Figure 135, and the definition of each field is explained below.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 290 of 1169

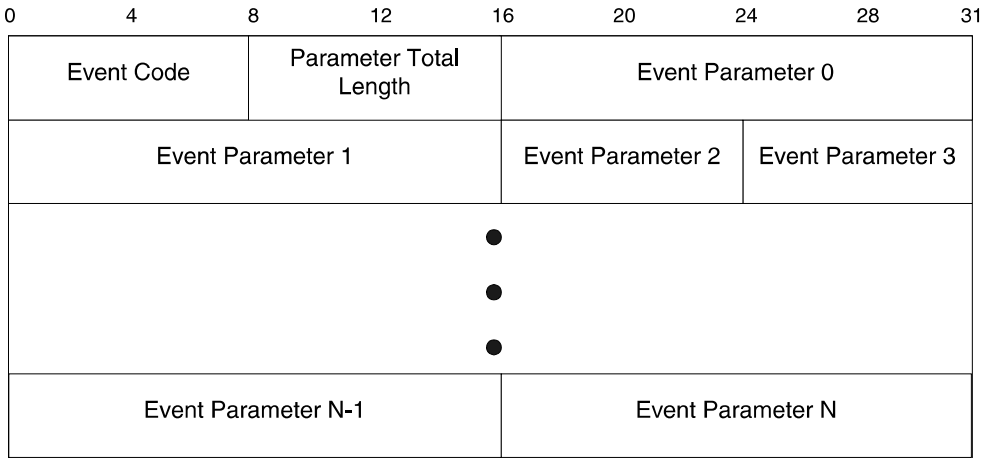


Figure 135—HCI event packet

Event_Code: *Size: 1 Byte*

Value	Parameter description
0xXX	Each event is assigned a 1-Byte event code used to uniquely identify different types of events. Range: 0x00–0xFF (The event code 0xFF is reserved for the event code used for vendor-specific debug events. In addition, the event code 0xFE is also reserved for Bluetooth Logo Testing).

Parameter_Total_Length: *Size: 1 Byte*

Value	Parameter description
0xXX	Length of all of the parameters contained in this packet, measured in bytes.

Event_Parameter 0–N: *Size: Parameter Total Length*

Value	Parameter description
0xXX	Each event has a specific number of parameters associated with it. These parameters and the size of each of the parameters are defined for each event. Each parameter is an integer number of bytes in size.

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 291 of 1169

11.2.4.3 HCI data packets

HCI Data Packets are used to exchange data between the Host and Host Controller. The data packets are defined for both ACL and SCO data types. The format of the HCI ACL Data Packet is shown in Figure 136, and the format of the SCO Data Packet is shown in Figure 137. The definition for each of the fields in the data packets is explained below.

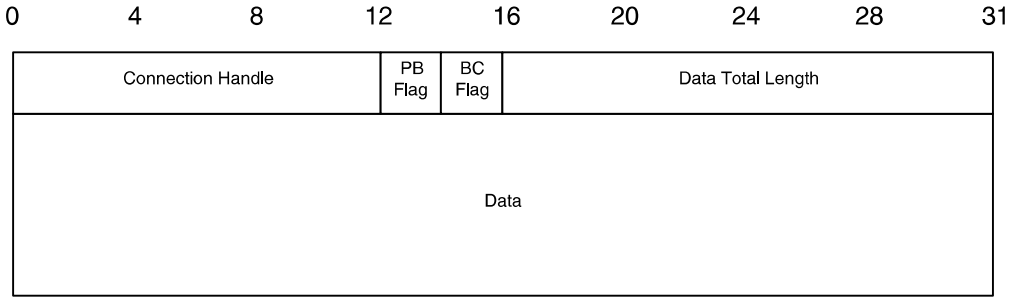


Figure 136—HCI ACL data packet

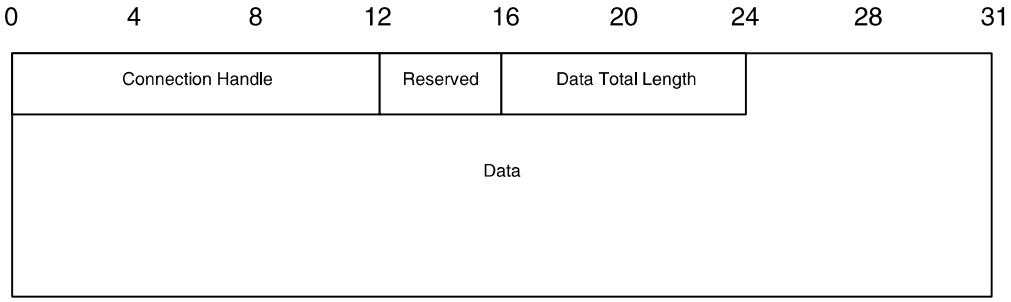


Figure 137—HCI SCO data packet

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 292 of 1169

Connection_Handle: *Size: 12 Bits*

Value	Parameter description
0xXXX	Connection Handle to be used for transmitting a data packet or segment. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use). The first time the Host sends an HCI Data Packet with Broadcast_Flag set to 01b (active broadcast) or 10b (piconet broadcast) after a power-on or a reset, the value of the Connection_Handle parameter shall be a value which is not currently assigned by the Host Controller. The Host shall use different connection handles for active broadcast and piconet broadcast. The Host Controller must then continue to use the same connection handles for each type of broadcast until a reset is made. Note: The Host Controller shall not send a Connection Complete event containing a new Connection_Handle that it knows is used for broadcast. Note: In some situations, it may happen that the Host Controller sends a Connection Complete event before having interpreted a Broadcast packet received from the Host, and that the Connection_Handles of both Connection Complete event and HCI Data packet are the same. This conflict has to be avoided as follows: If a Connection Complete event is received containing one of the connection handles used for broadcast, the Host has to wait before sending any packets for the new connection until it receives a Number Of Completed Packets event indicating that there are no pending broadcast packets belonging to the connection handle. In addition, the Host shall change the Connection_Handle used for the corresponding type of broadcast to a Connection_Handle which is currently not assigned by the Host Controller. This Connection_Handle must then be used for all the following broadcasts of that type until a reset is performed or the same conflict situation happens again. However, this will occur very rarely. The Host Controller shall, in the above conflict case, be able to distinguish between the Broadcast message sent by the Host and the new connection made (this could be even a new SCO link) even though the connection handles are the same. For an HCI Data Packet sent from the Host Controller to the Host where the Broadcast_Flag is 01 or 10, the Connection_Handle parameter should contain the connection handle for the ACL connection to the master that sent the broadcast. Note: Connection handles used for Broadcast do not identify an ACL point-to-point connection, so they shall not be used in any command having a Connection_Handle parameter and they will not be returned in any event having a Connection_Handle parameter except the Number Of Completed Packets event.

Flags: *Size: 2 Bits*

The Flag Bits consist of the Packet_Boundary_Flag and Broadcast_Flag. The Packet_Boundary_Flag is located in bit 4 and bit 5, and the Broadcast_Flag is located in bit 6 and 7 in the second byte of the HCI ACL Data packet.

Copyright © 2002 IEEE. All rights reserved.

Packet_Boundary_Flag: *Size: 2 Bits*

Value	Parameter description
00	Reserved for future use.
01	Continuing fragment packet of Higher Layer Message.
10	First packet of Higher Layer Message (i.e., start of an L2CAP packet).
11	Reserved for future use.

Broadcast_Flag (in packet from Host to Host Controller): *Size: 2 Bits*

Value	Parameter description
00	No broadcast. Only point-to-point.
01	Active Broadcast: packet is sent to all active slaves (i.e., packet is usually not sent during park beacon slots), and it may be received by slaves in sniff or park mode. See note below.
10	Piconet Broadcast: packet is sent to all slaves and all slaves in park mode (i.e. packet is sent during park beacon slots if there are parked slaves), and it may be received by slaves in sniff mode. See note below.
11	Reserved for future use.

Broadcast_Flag (in packet from Host Controller to Host): *Size: 2 Bits*

Value	Parameter description
00	Point-to-point.
01	Packet received as a slave not in park mode (either Active Broadcast or Piconet Broadcast).
10	Packet received as a slave in park mode (Piconet Broadcast).
11	Reserved for future use.

NOTE—Active broadcast packets may be sent in park beacon slots for synchronization since a slave can synchronize to any baseband packet that is preceded by the proper channel access code.

Slaves in sniff mode may or may not receive an active or piconet broadcast packet depending on whether they happen to be listening at sniff slots when the packet is sent.

Data_Total_Length: *Size: 2 Bytes*

Value	Parameter description
0xXXXX	Length of data measured in bytes.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 294 of 1169

Connection_Handle: *Size: 12 Bits*

Value	Parameter description
0xXXX	Connection handle to be used to for transmitting a SCO data packet or segment. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

The Reserved Bits consist of four bits which are located from bit 4 to bit 7 in the second byte of the HCI SCO Data packet.

Reserved: *Size: 4 Bits*

Value	Parameter description
XXXX	Reserved for future use.

Data_Total_Length: *Size: 1 Byte*

Value	Parameter description
0xXX	Length of SCO data measured in bytes.

11.2.5 Link Control Commands

The Link Control commands allow the Host Controller to control connections to other Bluetooth devices. When the Link Control commands are used, the LM controls how the Bluetooth piconets and scatternets are established and maintained. These commands instruct the LM to create and modify link layer connections with Bluetooth remote devices, perform Inquiries of other Bluetooth devices in range, and other LMP commands. For the Link Control commands, the OGF is defined as 0x01.

Command	Command summary description
Inquiry	The Inquiry command will cause the Bluetooth device to enter Inquiry Mode. Inquiry Mode is used to discovery other nearby Bluetooth devices.
Inquiry_Cancel	The Inquiry_Cancel command will cause the Bluetooth device to stop the current Inquiry if the Bluetooth device is in Inquiry Mode.
Periodic_Inquiry_Mode	The Periodic_Inquiry_Mode command is used to configure the Bluetooth device to perform an automatic Inquiry based on a specified period range.
Exit_Periodic_Inquiry_Mode	The Exit_Periodic_Inquiry_Mode command is used to end the Periodic Inquiry mode when the local device is in Periodic Inquiry Mode.
Create_Connection	The Create_Connection command will cause the link manager to create an ACL connection to the Bluetooth device with the BD_ADDR specified by the command parameters.

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 295 of 1169

Command	Command summary description
Disconnect	The Disconnect command is used to terminate an existing connection.
Add_SCO_Connection	The Add_SCO_Connection command will cause the link manager to create a SCO connection using the ACL connection specified by the Connection Handle command parameter.
Accept_Connection_Request	The Accept_Connection_Request command is used to accept a new incoming connection request.
Reject_Connection_Request	The Reject_Connection_Request command is used to decline a new incoming connection request.
Link_Key_Request_Reply	The Link_Key_Request_Reply command is used to reply to a Link Key Request event from the Host Controller, and specifies the Link Key stored on the Host to be used as the link key for the connection with the other Bluetooth device specified by BD_ADDR.
Link_Key_Request_Negative_Reply	The Link_Key_Request_Negative_Reply command is used to reply to a Link Key Request event from the Host Controller if the Host does not have a stored Link Key for the connection with the other Bluetooth Device specified by BD_ADDR.
PIN_Code_Request_Reply	The PIN_Code_Request_Reply command is used to reply to a PIN Code Request event from the Host Controller and specifies the PIN code to use for a connection.
PIN_Code_Request_Negative_Reply	The PIN_Code_Request_Negative_Reply command is used to reply to a PIN Code Request event from the Host Controller when the Host cannot specify a PIN code to use for a connection.
Change_Connection_Packet_Type	The Change_Connection_Packet_Type command is used to change which packet types can be used for a connection that is currently established.
Authentication_Requested	The Authentication_Requested command is used to establish authentication between the two devices associated with the specified Connection Handle.
Set_Connection_Encryption	The Set_Connection_Encryption command is used to enable and disable the link level encryption.
Change_Connection_Link_Key	The Change_Connection_Link_Key command is used to force both devices of a connection associated to the connection handle, to generate a new link key.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 296 of 1169

Command	Command summary description
Master_Link_Key	The Master_Link_Key command is used to force both devices of a connection associated to the connection handle to use the temporary link key of the Master device or the regular link keys.
Remote_Name_Request	The Remote_Name_Request command is used to obtain the user-friendly name of another Bluetooth device.
Read_Remote_Supported_Features	The Read_Remote_Supported_Features command requests a list of the supported features of a remote device.
Read_Remote_Version_Information	The Read_Remote_Version_Information command will read the values for the version information for the remote Bluetooth device.
Read_Clock_Offset	The Read_Clock_Offset command allows the Host to read the clock offset of remote devices.

11.2.5.1 Inquiry

Command	OCF	Command parameters	Return parameters
HCI_Inquiry	0x0001	LAP, Inquiry_Length, Num_Responses	—

Description:

This command will cause the Bluetooth device to enter Inquiry Mode. Inquiry Mode is used to discover other nearby Bluetooth devices. The LAP input parameter contains the LAP from which the inquiry access code shall be derived when the inquiry procedure is made. The Inquiry_Length parameter specifies the total duration of the Inquiry Mode and, when this time expires, Inquiry will be halted. The Num_Responses parameter specifies the number of responses that can be received before the Inquiry is halted. A Command Status event is sent from the Host Controller to the Host when the Inquiry command has been started by the Bluetooth device. When the Inquiry process is completed, the Host Controller will send an Inquiry Complete event to the Host indicating that the Inquiry has finished. The event parameters of Inquiry Complete event will have a summary of the result from the Inquiry process, which reports the number of nearby Bluetooth devices that responded. When a Bluetooth device responds to the Inquiry message, an Inquiry Result event will occur to notify the Host of the discovery.

A device that responds during an inquiry or inquiry period should always be reported to the Host in an Inquiry Result event if the device has not been reported earlier during the current inquiry or inquiry period and the device has not been filtered out using the command Set_Event_Filter. If the device has been reported earlier during the current inquiry or inquiry period, it may or may not be reported depending on the implementation (depending on if earlier results have been saved in the Host Controller and in that case how many responses that have been saved). It is recommended that the Host Controller tries to report a particular device only once during an inquiry or inquiry period.

Copyright © 2002 IEEE. All rights reserved.

Command Parameters:

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 2.4.3.

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 s Range: 1.28–61.44 s

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

Return Parameters:

None.

Event(s) generated (unless masked away):

A Command Status event is sent from the Host Controller to the Host when the Host Controller has started the Inquiry process. An Inquiry Result event will be created for each Bluetooth device that responds to the Inquiry message. In addition, multiple Bluetooth devices which respond to the Inquire message may be combined into the same event. An Inquiry Complete event is generated when the Inquiry process has completed.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Inquiry Complete event will indicate that this command has been completed. No Inquiry Complete event will be generated for the canceled Inquiry process.

11.2.5.2 Inquiry_Cancel

Command	OCF	Command parameters	Return parameters
HCI_Inquiry_Cancel	0x0002		Status

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 298 of 1169

Description:

This command will cause the Bluetooth device to stop the current Inquiry if the Bluetooth device is in Inquiry Mode. This command allows the Host to interrupt the Bluetooth device and request the Bluetooth device to perform a different task. The command should only be issued after the Inquiry command has been issued, a Command Status event has been received for the Inquiry command, and before the Inquiry Complete event occurs.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Inquiry_Cancel command succeeded.
0x01–0xFF	Inquiry_Cancel command failed. See Table 109 for list of Error Codes.

Event(s) generated (unless masked away):

When the Inquiry Cancel command has completed, a Command Complete event will be generated. No Inquiry Complete event will be generated for the canceled Inquiry process.

11.2.5.3 Periodic_Inquiry_Mode

Command	OCF	Command parameters	Return parameters
HCI_Periodic_ Inquiry_Mode	0x0003	Max_Period_Length, Min_Period_Length, LAP, Inquiry_Length, Num_Responses	Status

Description:

The Periodic_Inquiry_Mode command is used to configure the Bluetooth device to enter the Periodic Inquiry Mode that performs an automatic Inquiry. Max_Period_Length and Min_Period_Length define the time range between two consecutive inquiries, from the beginning of an inquiry until the start of the next inquiry. The Host Controller will use this range to determine a new random time between two consecutive inquiries for each Inquiry. The LAP input parameter contains the LAP from which the inquiry access code shall be derived when the inquiry procedure is made. The Inquiry_Length parameter specifies the total duration of the Inquiry Mode and, when time expires, Inquiry will be halted. The Num_Responses parameter specifies the number of responses that can be received before the Inquiry is halted. This command is completed when the Inquiry process has been started by the Bluetooth device, and a Command Complete event is sent from the Host Controller to the Host. When each of the periodic Inquiry processes are completed, the Host Controller will send an Inquiry Complete event to the Host indicating that the latest periodic Inquiry process has finished. The event parameters of Inquiry Complete event will have a summary of the result from the previous Periodic Inquiry process, which reports the number of nearby Bluetooth devices that responded. When a Bluetooth device responds to the Inquiry message an Inquiry Result event will occur to notify the Host of the discovery. Note that Max_Period_Length > Min_Period_Length > Inquiry_Length.

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 299 of 1169

A device which responds during an inquiry or inquiry period should always be reported to the Host in an Inquiry Result event if the device has not been reported earlier during the current inquiry or inquiry period and the device has not been filtered out using the command Set_Event_Filter. If the device has been reported earlier during the current inquiry or inquiry period, it may or may not be reported depending on the implementation (depending on if earlier results have been saved in the Host Controller and in that case how many responses that have been saved). It is recommended that the Host Controller tries to report a particular device only once during an inquiry or inquiry period.

Command Parameters:

Max_Period_Length: *Size: 2 Bytes*

Value	Parameter description
N = 0xXXXX	Maximum amount of time specified between consecutive inquiries. Size: 2 bytes Range: 0x03–0xFFFF Time = *N* * 1.28 s Range: 3.84–83884.8 s 0.0–23.3 h

Min_Period_Length: *Size: 2 Bytes*

Value	Parameter description
N = 0xXXXX	Minimum amount of time specified between consecutive inquiries. Size: 2 bytes Range: 0x02–0xFFFE Time = *N* * 1.28 s Range: 2.56–83883.52 s 0.0–23.3 h

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 2.4.3.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 300 of 1169

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 s Range: 1.28–61.44 s

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

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Periodic Inquiry Mode command succeeded.
0x01–0xFF	Periodic Inquiry Mode command failed. See Table 109 for list of Error Codes.

Event(s) generated (unless masked away):

The Periodic Inquiry Mode begins when the Host Controller sends the Command Complete event for this command to the Host. An Inquiry Result event will be created for each Bluetooth device that responds to the Inquiry message. In addition, multiple Bluetooth devices that respond to the Inquiry message may be combined into the same event. An Inquiry Complete event is generated when each of the periodic Inquiry processes has completed. No Inquiry Complete event will be generated for the canceled Inquiry process.

11.2.5.4 Exit_Periodic_Inquiry_Mode

Command	OCF	Command Parameters	Return parameters
HCI_Exit_Periodic_Inquiry_Mode	0x0004		Status

Description:

The Exit Periodic Inquiry Mode command is used to end the Periodic Inquiry mode when the local device is in Periodic Inquiry Mode. If the local device is currently in an Inquiry process, the Inquiry process will be stopped directly and the Host Controller will no longer perform periodic inquiries until the Periodic Inquiry Mode command is reissued.

Copyright © 2002 IEEE. All rights reserved.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Exit Periodic Inquiry Mode command succeeded.
0x01–0xFF	Exit Periodic Inquiry Mode command failed. See Table 109 for list of Error Codes.

Event(s) generated (unless masked away):

A Command Complete event for this command will occur when the local device is no longer in Periodic Inquiry Mode. No Inquiry Complete event will be generated for the canceled Inquiry process.

11.2.5.5 Create_Connection

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	

Description:

This command will cause the Link Manager to create a connection to the Bluetooth device with the BD_ADDR specified by the command parameters. This command causes the local Bluetooth device to begin the Page process to create a link level connection. The Link Manager will determine how the new ACL connection is established. This ACL connection is determined by the current state of the device, its piconet, and the state of the device to be connected. The Packet_Type command parameter specifies which packet types the Link Manager shall use for the ACL connection. The Link Manager must use only the packet type(s) specified by the Packet_Type command parameter for sending HCI ACL Data Packets. Multiple packet types may be specified for the Packet Type parameter by performing a bit-wise OR operation of the different packet types. The Link Manager may choose which packet type to be used from the list of acceptable packet types. The Page_Scan_Repetition_Mode and Page_Scan_Mode parameters specify the page scan modes supported by the remote device with the BD_ADDR. This is the information that was acquired during the inquiry process. The Clock_Offset parameter is the difference between its own clock and the clock of the remote device with BD_ADDR. Only bits 2 through 16 of the difference are used, and they are mapped to this parameter as bits 0 through 14, respectively. A Clock_Offset_Valid_Flag, located in bit 15 of the Clock_Offset parameter, is used to indicate if the Clock Offset is valid or not. A Connection handle for this connection is returned in the Connection Complete event (see below). The Allow_Role_Switch parameter specifies if the local device accepts or rejects the request of a master-slave role switch when the remote device requests it at the connection setup (in the Role parameter of the

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 302 of 1169

Accept_Connection_Request command) (before the local Host Controller returns a Connection Complete event). For a definition of the different packet types, see Clause 8.

Note that at least one packet type shall be specified. The Host should enable as many packet types as possible for the Link Manager to perform efficiently. However, the Host shall not enable packet types that the local device does not support.

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

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

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

Copyright © 2002 IEEE. All rights reserved.

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.

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

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.

Return Parameters:

None.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 304 of 1169

Event(s) generated (unless masked away):

When the Host Controller receives the Create Connection command, the Host Controller sends the Command Status event to the Host. In addition, when the LM determines the connection is established, the Host Controller, on both Bluetooth devices that form the connection, will send a Connection Complete event to each Host. The Connection Complete event contains the Connection Handle if this command is successful.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Connection Complete event will indicate that this command has been completed.

11.2.5.6 Disconnect

Command	OCF	Command parameters	Return parameters
HCI_Disconnect	0x0006	Connection_Handle, Reason	

Description:

The Disconnection command is used to terminate an existing connection. The Connection_Handle command parameter indicates which connection is to be disconnected. The Reason command parameter indicates the reason for ending the connection. The remote Bluetooth device will receive the Reason command parameter in the Disconnection Complete event. All SCO connections on a physical link should be disconnected before the ACL connection on the same physical connection is disconnected.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle for the connection being disconnected. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Reason: *Size: 1 Byte*

Value	Parameter description
0x05, 0x13–0x15, 0x1A, 0x29	Authentication Failure error code (0x05), Other End Terminated Connection error codes (0x13–0x15). Unsupported Remote Feature error code (0x1A) and Pairing with Unit Key Not Supported error code (0x29). See Table 109 for list of Error Codes.

Return Parameters:

None.

Copyright © 2002 IEEE. All rights reserved.

Event(s) generated (unless masked away):

When the Host Controller receives the Disconnect command, it sends the Command Status event to the Host. The Disconnection Complete event will occur at each Host when the termination of the connection has completed, and indicates that this command has been completed.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Disconnection Complete event will indicate that this command has been completed.

11.2.5.7 Add_SCO_Connection

Command	OCF	Command parameters	Return parameters
HCI_Add_SCO_Connection	0x0007	Connection_Handle, Packet_Type	—

Description:

This command will cause the link manager to create a SCO connection using the ACL connection specified by the Connection_Handle command parameter. This command causes the local Bluetooth device to create a SCO connection. The Link Manager will determine how the new connection is established. This connection is determined by the current state of the device, its piconet, and the state of the device to be connected. The Packet_Type command parameter specifies which packet types the Link Manager should use for the connection. The Link Manager must only use the packet type(s) specified by the Packet_Type command parameter for sending HCI SCO Data Packets. Multiple packet types may be specified for the Packet_Type command parameter by performing a bitwise OR operation of the different packet types. The Link Manager may choose which packet type is to be used from the list of acceptable packet types. A Connection Handle for this connection is returned in the Connection Complete event (see below).

Note that an SCO connection can only be created when an ACL connection already exists and when it is not put in park mode. For a definition of the different packet types, see Clause 8.

Note also that at least one packet type shall be specified. The Host should enable as many packet types as possible for the Link Manager to perform efficiently. However, the Host shall not enable packet types that the local device does not support.

Command Parameters:

Connection_Handle *Size 2 Bytes (12 Bits meaningful)*

Value	Parameter description
0xXXXX	Connection Handle for the ACL connection being used to create an SCO connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 306 of 1169

Packet_Type: *Size: 2 Bytes*

Value	Parameter description
0x0001	Reserved for future use.
0x0002	Reserved for future use.
0x0004	Reserved for future use.
0x0008	Reserved for future use.
0x0010	Reserved for future use.
0x0020	HV1
0x0040	HV2
0x0080	HV3
0x0100	Reserved for future use.
0x0200	Reserved for future use.
0x0400	Reserved for future use.
0x0800	Reserved for future use.
0x1000	Reserved for future use.
0x2000	Reserved for future use.
0x4000	Reserved for future use.
0x8000	Reserved for future use.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Add_SCO_Connection command, it sends the Command Status event to the Host. In addition, when the LM determines the connection is established, the Host Controller, on both Bluetooth devices that form the connection, will send a Connection Complete event to each Host. The Connection Complete event contains the Connection Handle if this command is successful. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Connection Complete event will indicate that this command has been completed.

11.2.5.8 Accept_Connection_Request

Command	OCF	Command parameters	Return parameters
HCI_Accept_Connection Request	0x0009	BD_ADDR, Role	—

Copyright © 2002 IEEE. All rights reserved.

Description:

The Accept_Connection_Request command is used to accept a new incoming connection request. The Accept_Connection_Request command shall only be issued after a Connection Request event has occurred. The Connection Request event will return the BD_ADDR of the device that is requesting the connection. This command will cause the Link Manager to create a connection to the Bluetooth device, with the BD_ADDR specified by the command parameters. The Link Manager will determine how the new connection will be established. This will be determined by the current state of the device, its piconet, and the state of the device to be connected. The Role command parameter allows the Host to specify if the Link Manager shall perform a Master-Slave switch, and become the Master for this connection. Also, the decision to accept a connection shall be completed before the connection accept timeout expires on the local Bluetooth Module.

Note that when accepting an SCO connection request, the Role parameter is not used and will be ignored by the Host Controller.

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

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

Role: *Size: 1 Byte*

Value	Parameter description
0x00	Become the Master for this connection. The LM will perform the Master/Slave switch.
0x01	Remain the Slave for this connection. The LM will **NOT** perform the Master/Slave switch.

Return Parameters:

None.

Event(s) generated (unless masked away):

The Accept_Connection_Request command will cause the Command Status event to be sent from the Host Controller when the Host Controller begins setting up the connection. In addition, when the Link Manager determines the connection is established, the Host Controllers on both Bluetooth devices that form the connection will send a Connection Complete event to each Host. The Connection Complete event contains the Connection Handle if this command is successful.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Connection Complete event will indicate that this command has been completed.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 308 of 1169

11.2.5.9 Reject_Connection_Request

Command	OCF	Command parameters	Return parameters
HCI_Reject_Connection_Request	0x000A	BD_ADDR, Reason	—

Description:

The Reject_Connection_Request command is used to decline a new incoming connection request. The Reject_Connection_Request command shall only be called after a Connection Request event has occurred. The Connection Request event will return the BD_ADDR of the device that is requesting the connection. The Reason command parameter will be returned to the connecting device in the Status parameter of the Connection Complete event returned to the Host of the connection device, to indicate why the connection was declined.

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device to reject the connection from.

Reason: *Size: 1 Byte*

Value	Parameter description
0x0D-0x0F	Host Reject Error Code. See Table 109 for list of Error Codes and descriptions.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Reject_Connection_Request command, the Host Controller sends the Command Status event to the Host. A Connection Complete event will then be sent, both to the local Host and to the Host of the device attempting to make the connection. The Status parameter of the Connection Complete event, which is sent to the Host of the device attempting to make the connection, will contain the Reason command parameter from this command.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Connection Complete event will indicate that this command has been completed.

Copyright © 2002 IEEE. All rights reserved.

11.2.5.10 Link_Key_Request_Reply

Command	OCF	Command parameters	Return parameters
HCI_Link_Key_Request_Reply	0x000B	BD_ADDR, Link_Key	Status, BD_ADDR

Description:

The Link_Key_Request_Reply command is used to reply to a Link Key Request event from the Host Controller, and specifies the Link Key stored on the Host to be used as the link key for the connection with the other Bluetooth Device specified by BD_ADDR. The Link Key Request event will be generated when the Host Controller needs a Link Key for a connection.

When the Host Controller generates a Link Key Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host shall respond with either a Link_Key_Request_Reply or Link_Key_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout (see Clause 9).

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device of which the Link Key is for.

Link_Key: *Size: 16 Bytes*

Value	Parameter description
0xXXXXXXXXXX XXXXXXXXXX XXXXXXXXXX	Link Key for the associated BD_ADDR.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Link_Key_Request_Reply command succeeded.
0x01–0xFF	Link_Key_Request_Reply command failed. See Table 109 for list of Error Codes.

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXX XXXX	BD_ADDR of the Device of which the Link Key request reply has completed.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 310 of 1169

Event(s) generated (unless masked away):

The Link_Key_Request_Reply command will cause a Command Complete event to be generated.

11.2.5.11 Link_Key_Request_Negative_Reply

Command	OCF	Command parameters	Return parameters
HCI_Link_Key_Request_Negative_Reply	0x000C	BD_ADDR	Status, BD_ADDR

Description:

The Link_Key_Request_Negative_Reply command is used to reply to a Link Key Request event from the Host Controller if the Host does not have a stored Link Key for the connection with the other Bluetooth Device specified by BD_ADDR. The Link Key Request event will be generated when the Host Controller needs a Link Key for a connection.

When the Host Controller generates a Link Key Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host shall respond with either a Link_Key_Request_Reply or Link_Key_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout (see Clause 9).

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device which the Link Key is for.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Link_Key_Request_Negative_Reply command succeeded.
0x01–0xFF	Link_Key_Request_Negative_Reply command failed. See Table 109 for list of Error Codes.

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device which the Link Key request negative reply has completed.

Event(s) generated (unless masked away):

The Link_Key_Request_Negative_Reply command will cause a Command Complete event to be generated.

Copyright © 2002 IEEE. All rights reserved.

11.2.5.12 PIN_Code_Request_Reply

Command	OCF	Command parameters	Return parameters
HCI_PIN_Code_Request_Reply	0x000D	BD_ADDR, PIN_Code_Length, PIN_Code	Status, BD_ADDR

Description:

The PIN_Code_Request_Reply command is used to reply to a PIN Code request event from the Host Controller, and specifies the PIN code to use for a connection. The PIN Code Request event will be generated when a connection with remote initiating device has requested pairing.

When the Host Controller generates a PIN Code Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host shall respond with either a PIN_Code_Request_Reply or PIN_Code_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout (see Clause 9).

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device which the PIN code is for.

PIN_Code_Length: *Size: 1 Byte*

Value	Parameter description
0xXX	The PIN code length specifics the length, in bytes, of the PIN code to be used. Range: 0x01–0x10

PIN_Code: *Size: 16 Bytes*

Value	Parameter description
0xXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	PIN code for the device that is to be connected. The Host should insure that strong PIN Codes are used. PIN Codes can be up to a maximum of 128 bits. Note: the PIN_Code Parameter is a string parameter. Endianess does therefore not apply to the PIN_Code Parameter. The first byte of the PIN code should be transmitted first.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 312 of 1169

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	PIN_Code_Request_Reply command succeeded.
0x01-0xFF	PIN_Code_Request_Reply command failed. See Table 109 for list of Error Codes.

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device which the PIN Code request reply has completed.

Event(s) generated (unless masked away):

The PIN_Code_Request_Reply command will cause a Command Complete event to be generated.

11.2.5.13 PIN_Code_Request_Negative_Reply

Command	OCF	Command parameters	Return parameters
HCI_PIN_Code_ Request_Negative Reply	0x000E	BD_ADDR	Status, BD_ADDR

Description:

The PIN_Code_Request_Negative_Reply command is used to reply to a PIN Code request event from the Host Controller when the Host cannot specify a PIN code to use for a connection. This command will cause the pair request with remote device to fail.

When the Host Controller generates a PIN Code Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host shall respond with either a PIN_Code_Request_Reply or PIN_Code_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout. (See Clause 9.)

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device which this command is responding to.

Copyright © 2002 IEEE. All rights reserved.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	PIN_Code_Request_Negative_Reply command succeeded.
0x01–0xFF	PIN_Code_Request_Negative_Reply command failed. See Table 109 for list of Error Codes.

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device which the PIN Code request negative reply has completed.

Event(s) generated (unless masked away):

The PIN_Code_Request_Negative_Reply command will cause a Command Complete event to be generated.

11.2.5.14 Change_Connection_Packet_Type

Command	OCF	Command parameters	Return parameters
HCI_Change_Connection_Packet_Type	0x000F	Connection_Handle, Packet_Type	

Description:

The Change_Connection_Packet_Type command is used to change which packet types can be used for a connection that is currently established. This allows current connections to be dynamically modified to support different types of user data. The Packet_Type command parameter specifies which packet types the Link Manager can use for the connection. The Link Manager must only use the packet type(s) specified by the Packet_Type command parameter for sending HCI Data Packets. The interpretation of the value for the Packet_Type command parameter will depend on the Link_Type command parameter returned in the Connection Complete event at the connection setup. Multiple packet types may be specified for the Packet_Type command parameter by bitwise OR operation of the different packet types. For a definition of the different packet types, see Clause 8.

Note that at least one packet type shall be specified. The Host should enable as many packet types as possible for the Link Manager to perform efficiently. However, the Host shall not enable packet types that the local device does not support.

Copyright © 2002 IEEE. All rights reserved.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to for transmitting and receiving voice or data. Returned from creating a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Packet_Type: *Size: 2 Bytes*

For ACL Link_Type

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

Copyright © 2002 IEEE. All rights reserved.

For SCO Link Type

Value	Parameter description
0x0001	Reserved for future use.
0x0002	Reserved for future use.
0x0004	Reserved for future use.
0x0008	Reserved for future use.
0x0010	Reserved for future use.
0x0020	HV1
0x0040	HV2
0x0080	HV3
0x0100	Reserved for future use.
0x0200	Reserved for future use.
0x0400	Reserved for future use.
0x0800	Reserved for future use.
0x1000	Reserved for future use.
0x2000	Reserved for future use.
0x4000	Reserved for future use.
0x8000	Reserved for future use.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Change Connection Packet Type command, the Host Controller sends the Command Status event to the Host. In addition, when the Link Manager determines the packet type has been changed for the connection, the Host Controller on the local device will send a Connection Packet Type Changed event to the Host. This will be done at the local side only. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Connection Packet Type Changed event will indicate that this command has been completed.

11.2.5.15 Authentication_Requested

Command	OCF	Command parameters	Return parameters
HCI_Authentication_ Requested	0x0011	Connection_Handle	—

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 316 of 1169

Description:

The Authentication_Requested command is used to try to authenticate the remote device associated with the specified Connection Handle. The Host shall not issue the Authentication_Requested command with a Connection_Handle corresponding to an encrypted link. On an authentication failure, the Host Controller or Link Manager shall not automatically detach the link. The Host is responsible for issuing a Disconnect command to terminate the link if the action is appropriate. Note that the Connection_Handle command parameter is used to identify the other Bluetooth device, which forms the connection. The Connection Handle should be a Connection Handle for an ACL connection.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to set up authentication for all Connection Handles with the same Bluetooth device end-point as the specified Connection Handle. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Authentication_Requested command, it sends the Command Status event to the Host. The Authentication Complete event will occur when the authentication has been completed for the connection and is indication that this command has been completed. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Authentication Complete event will indicate that this command has been completed.

Note that when the local or remote Host Controller does not have a link key for the specified Connection_Handle, it will request the link key from its Host, before the local Host finally receives the Authentication Complete event.

11.2.5.16 Set_Connection_Encryption

Command	OCF	Command parameters	Return parameters
HCI_Set_Connection_ Encryption	0x0013	Connection_Handle, Encryption_Enable	—

Description:

The Set_Connection_Encryption command is used to enable and disable the link level encryption. Note that the Connection_Handle command parameter is used to identify the other Bluetooth device which forms the connection. The Connection Handle should be a Connection Handle for an ACL connection. While the encryption is being changed, all ACL traffic shall be turned off for all Connection Handles associated with the remote device.

Copyright © 2002 IEEE. All rights reserved.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to enable/disable the link layer encryption for all Connection Handles with the same Bluetooth device end-point as the specified Connection Handle. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Encryption_Enable: *Size: 1 Byte*

Value	Parameter description
0x00	Turn Link Level Encryption OFF.
0x01	Turn Link Level Encryption ON.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Set_Connection_Encryption command, the Host Controller sends the Command Status event to the Host. When the Link Manager has completed enabling/disabling encryption for the connection, the Host Controller on the local Bluetooth device will send an Encryption Change event to the Host, and the Host Controller on the remote device will also generate an Encryption Change event. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Encryption Change event will indicate that this command has been completed.

11.2.5.17 Change_Connection_Link_Key

Command	OCF	Command parameters	Return parameters
HCI_Change_Connection_Link_Key	0x0015	Connection_Handle	—

Description:

The Change_Connection_Link_Key command is used to force both devices of a connection associated with the connection handle to generate a new link key. The link key is used for authentication and encryption of connections.

Note that the Connection_Handle command parameter is used to identify the other Bluetooth device forming the connection. The Connection Handle should be a Connection Handle for an ACL connection. If the connection encryption is enabled, and the temporary link key is currently used, then the Bluetooth master device will automatically restart the encryption.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 318 of 1169

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Change_Connection_Link_Key command, the Host Controller sends the Command Status event to the Host. When the Link Manager has changed the Link Key for the connection, the Host Controller on the local Bluetooth device will send a Link Key Notification event and a Change Connection Link Key Complete event to the Host, and the Host Controller on the remote device will also generate a Link Key Notification event. The Link Key Notification event indicates that a new connection link key is valid for the connection.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Change Connection Link Key Complete event will indicate that this command has been completed.

11.2.5.18 Master_Link_Key

Command	OCF	Command parameters	Return parameters
HCI_Master_Link_Key	0x0017	Key_Flag	—

Description:

The Master Link Key command is used to force the device that is master of the piconet to use the temporary link key of the master device, or the semipermanent link keys. The temporary link key is used for encryption of broadcast messages within a piconet, and the semipermanent link keys are used for private encrypted point-to-point communication. The Key_Flag command parameter is used to indicate which Link Key (temporary link key of the Master, or the semipermanent link keys) shall be used.

Command Parameters:

Key_Flag: *Size: 1 Byte*

Value	Parameter description
0x00	Use semipermanent Link Keys.
0x01	Use Temporary Link Key.

Return Parameters:

None.

Copyright © 2002 IEEE. All rights reserved.

Event(s) generated (unless masked away):

When the Host Controller receives the Master_Link_Key command, the Host Controller sends the Command Status event to the Host. When the Link Manager has changed link key, the Host Controller on both the local and the remote device will send a Master Link Key Complete event to the Host. The Connection Handle on the master side will be a Connection Handle for one of the existing connections to a slave. On the slave side, the Connection Handle will be a Connection Handle to the initiating master.

The Master Link Key Complete event contains the status of this command.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Master Link Key Complete event will indicate that this command has been completed.

11.2.5.19 Remote_Name_Request

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

Description:

The Remote_Name_Request command is used to obtain the user-friendly name of another Bluetooth device. The user-friendly name is used to enable the user to distinguish one Bluetooth device from another. The BD_ADDR command parameter is used to identify the device for which the user-friendly name is to be obtained. The Page_Scan_Repetition_Mode and Page_Scan_Mode command parameters specify the page scan modes supported by the remote device with the BD_ADDR. This is the information that was acquired during the inquiry process. The Clock_Offset parameter is the difference between its own clock and the clock of the remote device with BD_ADDR. Only bits 2 through 16 of the difference are used and they are mapped to this parameter as bits 0 through 14, respectively. A Clock_Offset_Valid_Flag, located in bit 15 of the Clock_Offset command parameter, is used to indicate if the Clock Offset is valid or not.

Note that if no connection exists between the local device and the device corresponding to the BD_ADDR, a temporary link layer connection will be established to obtain the name of the remote device.

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

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

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 320 of 1169

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.

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

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Remote_Name_Request command, the Host Controller sends the Command Status event to the Host. When the Link Manager has completed the LMP messages to obtain the remote name, the Host Controller on the local Bluetooth device will send a Remote Name Request Complete event to the Host. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, only the Remote Name Request Complete event will indicate that this command has been completed.

Copyright © 2002 IEEE. All rights reserved.

11.2.5.20 Read_Remote_Supported_Features

Command	OCF	Command parameters	Return parameters
HCI_Read_Remote_Supported_Features	0x001B	Connection_Handle	—

Description:

This command requests a list of the supported features for the remote device identified by the Connection_Handle parameter. The Connection_Handle shall be a Connection_Handle for an ACL connection. The Read Remote Supported Features Complete event will return a list of the LMP features. For details, see Clause 9.

Command Parameters:

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

Value	Parameter description
0xXXXX	Specifies which Connection Handle's LMP-supported features list to get. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Read_Remote_Supported_Features command, the Host Controller sends the Command Status event to the Host. When the Link Manager has completed the LMP messages to determine the remote features, the Host Controller on the local Bluetooth device will send a Read Remote Supported Features Complete event to the Host. The Read Remote Supported Features Complete event contains the status of this command, and parameters describing the supported features of the remote device. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Read Remote Supported Features Complete event will indicate that this command has been completed.

11.2.5.21 Read_Remote_Version_Information

Command	OCF	Command parameters	Return parameters
HCI_Read_Remote_Version_ Information	0x001D	Connection_Handle	—

Description:

This command will obtain the values for the version information for the remote Bluetooth device identified by the Connection_Handle parameter. The Connection_Handle shall be a Connection_Handle for an ACL connection.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 322 of 1169

Command Parameters:

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

Value	Parameter description
0xXXXX	Specifies which Connection Handle's version information to get. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Read_Remote_Version_Information command, the Host Controller sends the Command Status event to the Host. When the Link Manager has completed the LMP messages to determine the remote version information, the Host Controller on the local Bluetooth device will send a Read Remote Version Information Complete event to the Host. The Read Remote Version Information Complete event contains the status of this command, and parameters describing the version and subversion of the LMP used by the remote device.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Read Remote Version Information Complete event will indicate that this command has been completed.

11.2.5.22 Read_Clock_Offset

Command	OCF	Command parameters	Return parameters
HCI_Read_Clock_Offset	0x001F	Connection_Handle	—

Description:

Both the System Clock and the clock offset to a remote device are used to determine what hopping frequency is used by a remote device for page scan. This command allows the Host to read clock offset to remote devices. The clock offset can be used to speed up the paging procedure when the local device tries to establish a connection to a remote device, for example, when the local Host has issued Create_Connection or Remote_Name_Request. The Connection_Handle shall be a Connection_Handle for an ACL connection.

Command Parameters:

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

Value	Parameter description
0xXXXX	Specifies which Connection Handle's Clock Offset parameter is returned. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Copyright © 2002 IEEE. All rights reserved.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the Read_Clock_Offset command, the Host Controller sends the Command Status event to the Host. If this command was requested at the master and the Link Manager has completed the LMP messages to obtain the Clock Offset information, the Host Controller on the local Bluetooth device will send a Read Clock Offset Complete event to the Host. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, only the Read Clock Offset Complete event will indicate that this command has been completed. If the command is requested at the slave, the LM will immediately send a Command Status event and a Read Clock Offset Complete event to the Host, without an exchange of LMP PDU.

11.2.6 Link Policy Commands

The Link Policy Commands provide methods for the Host to affect how the Link Manager manages the piconet. When Link Policy Commands are used, the LM still controls how Bluetooth piconets and scatternets are established and maintained, depending on adjustable policy parameters. These policy commands modify the Link Manager behavior that can result in changes to the link layer connections with Bluetooth remote devices.

Note that only one ACL connection can exist between two Bluetooth Devices, and therefore there can only be one ACL HCI Connection Handle for each physical link layer Connection. The Bluetooth Host Controller provides policy adjustment mechanisms to provide support for a number of different policies. This capability allows one Bluetooth module to be used to support many different usage models, and the same Bluetooth module can be incorporated in many different types of Bluetooth devices. For the Link Policy Commands, the OGF is defined as 0x02.

Command	Command summary description
Hold_Mode	The Hold_Mode command is used to alter the behavior of the LM and have the LM place the local or remote device into the hold mode.
Sniff_Mode	The Sniff_Mode command is used to alter the behavior of the LM and have the LM place the local or remote device into the sniff mode.
Exit_Sniff_Mode	The Exit_Sniff_Mode command is used to end the sniff mode for a connection handle which is currently in sniff mode.
Park_Mode	The Park_Mode command is used to alter the behavior of the LM and have the LM place the local or remote device into the Park mode.
Exit_Park_Mode	The Exit_Park_Mode command is used to switch the Bluetooth device from park mode back to active mode.
QoS_Setup	The QoS_Setup command is used to specify Quality of Service parameters for a connection handle.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 324 of 1169

Command	Command summary description
Role_Discovery	The Role_Discovery command is used for a Bluetooth device to determine which role the device is performing for a particular Connection Handle.
Switch_Role	The Switch_Role command is used for a Bluetooth device switch the current role the device is performing for a particular connection with the specified Bluetooth device
Read_Link_Policy_Settings	The Read_Link_Policy_Settings command will read the Link Policy settings for the specified Connection Handle. The Link Policy settings allow the Host to specify which Link Modes the LM can use for the specified Connection Handle.
Write_Link_Policy_Settings	The Write_Link_Policy_Settings command will write the Link Policy settings for the specified Connection Handle. The Link Policy settings allow the Host to specify which Link Modes the LM can use for the specified Connection Handle.

11.2.6.1 Hold_Mode

Command	OCF	Command parameters	Return parameters
HCI_Hold_Mode	0x0001	Connection_Handle, Hold_Mode_Max_Interval, Hold_Mode_Min_Interval	—

Description:

The Hold_Mode command is used to alter the behavior of the Link Manager, and have it place the ACL baseband connection associated by the specified Connection Handle into the hold mode. The Hold_Mode_Max_Interval and Hold_Mode_Min_Interval command parameters specify the length of time the Host wants to put the connection into the hold mode. The local and remote devices will negotiate the length in the hold mode. The Hold_Mode_Max_Interval parameter is used to specify the maximum length of the Hold interval for which the Host may actually enter into the hold mode after negotiation with the remote device. The Hold interval defines the amount of time between when the Hold Mode begins and when the Hold Mode is completed. The Hold_Mode_Min_Interval parameter is used to specify the minimum length of the Hold interval for which the Host may actually enter into the hold mode after the negotiation with the remote device. Therefore the Hold_Mode_Min_Interval cannot be greater than the Hold_Mode_Max_Interval. The Host Controller will return the actual Hold interval in the Interval parameter of the Mode Change event, if the command is successful. This command enables the Host to support a low-power policy for itself or several other Bluetooth devices, and allows the devices to enter Inquiry Scan, Page Scan, and a number of other possible actions.

Note that the connection handle cannot be of the SCO link type If the Host sends data to the Host Controller with a Connection_Handle corresponding to a connection in hold mode, the Host Controller will keep the data in its buffers until either the data can be transmitted (the hold mode has ended) or a flush, a flush

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 325 of 1169

timeout or a disconnection occurs. This is valid even if the Host has not yet been notified of the hold mode through a Mode Change event when it sends the data.

Note that the above is not valid for an HCI Data Packet sent from the Host to the Host Controller on the master side where the Connection_Handle is a Connection_Handle used for broadcast and the Broadcast_Flag is set to Active Broadcast or Piconet Broadcast. The broadcast data will then never be received by slaves in hold mode.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Hold_Mode_Max_Interval: *Size: 2 Bytes*

Value	Parameter description
N = 0xXXXX	Maximum acceptable number of Baseband slots to wait in Hold Mode. Time Length of the Hold = *N* * 0.625 ms (1 Baseband slot) Range for *N*: 0x0001–0xFFFF Time Range: 0.625 ms–40.9 s

Hold_Mode_Min_Interval: *Size: 2 Bytes*

Value	Parameter description
N = 0xXXXX	Minimum acceptable number of Baseband slots to wait in Hold Mode. Time Length of the Hold = *N* * 0.625 msec (1 Baseband slot) Range for *N*: 0x0001–0xFFFF Time Range: 0.625 ms–40.9 s

Return Parameters:

None.

Event(s) generated (unless masked away):

The Host Controller sends the Command Status event for this command to the Host when it has received the Hold_Mode command. The Mode Change event will occur when the Hold Mode has started and the Mode Change event will occur again when the Hold Mode has completed for the specified connection handle. The Mode Change event signaling the end of the Hold Mode is an estimation of the hold mode ending if the event is for a remote Bluetooth device.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, only the Mode Change event will indicate that this command has been completed. If an error occurs after the Command Status event has occurred, then the status in the Mode Change event will indicate the error.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 326 of 1169

11.2.6.2 Sniff_Mode

Command	OCF	Command parameters	Return parameters
HCI_Sniff_Mode	0x0003	Connection_Handle, Sniff_Max_Interval, Sniff_Min_Interval, Sniff_Attempt, Sniff_Timeout	—

Description:

The Sniff Mode command is used to alter the behavior of the Link Manager and have it place the ACL base-band connection associated with the specified Connection Handle into the sniff mode. The Connection_Handle command parameter is used to identify which ACL link connection is to be placed in sniff mode. The Sniff_Max_Interval and Sniff_Min_Interval command parameters are used to specify the requested acceptable maximum and minimum periods in the Sniff Mode. The Sniff_Min_Interval cannot be greater than the Sniff_Max_Interval. The sniff interval defines the amount of time between each consecutive sniff period. The Host Controller will return the actual sniff interval in the Interval parameter of the Mode Change event, if the command is successful. For a description of the meaning of the Sniff_Attempt and Sniff_Timeout parameters, see 8.10.8.2. Sniff_Attempt is there called $N_{\text{sniff attempt}}$ and Sniff_Timeout is called $N_{\text{sniff timeout}}$. This command enables the Host to support a low-power policy for itself or several other Bluetooth devices, and allows the devices to enter Inquiry Scan, Page Scan, and a number of other possible actions.

Note that in addition, the connection handle cannot be one of SCO link type. If the Host sends data to the Host Controller with a Connection_Handle corresponding to a connection in sniff mode, the Host Controller will keep the data in its buffers until either the data can be transmitted or a flush, a flush timeout or a disconnection occurs. This is valid even if the Host has not yet been notified of the sniff mode through a Mode Change event when it sends the data. Note that it is possible for the master to transmit data to a slave without exiting sniff mode (see description in 10.8.2).

Note that the above is not valid for an HCI Data Packet sent from the Host to the Host Controller on the master side where the Connection_Handle is a Connection_Handle used for broadcast and the Broadcast_Flag is set to Active Broadcast or Piconet Broadcast. In that case, the broadcast data will only be received by a slave in sniff mode if that slave happens to listen to the master when the broadcast is made.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 327 of 1169

Sniff_Max_Interval: *Size: 2 Byte*

Value	Parameter description
N = 0xXXXX	Maximum acceptable number of Baseband slots between each sniff period. (Sniff_Max_Interval >= Sniff_Min_Interval) Length = *N* * 0.625 ms (1 Baseband slot) Range for *N*: 0x0001–0xFFFF Time Range: 0.625 ms–40.9 s

Sniff_Min_Interval: *Size: 2 Byte*

Value	Parameter description
N = 0xXXXX	Minimum acceptable number of Baseband slots between each sniff period. (Sniff_Max_Interval >= Sniff_Min_Interval) Length = *N* * 0.625 ms (1 Baseband slot) Range for *N*: 0x0001–0xFFFF Time Range: 0.625 ms–40.9 s

Sniff_Attempt: *Size: 2 Byte*

Value	Parameter description
N = 0xXXXX	Number of Baseband receive slots for sniff attempt. Length = (2 * *N* – 1)* 0.625 ms Range for *N*: 0x0001–0x7FFF Time Range: 0.625 ms–40.9 s

Sniff_Timeout: *Size: 2 Byte*

Value	Parameter description
N = 0xXXXX	Number of Baseband recieve slots for sniff timeout. Length = (2 **N* – 1) * 0.625 ms if *N* > 0, Length = 0 if *N* = 0 Range for *N*: 0x0000–0x7FFF Time Range: 0 ms–40.9 s

Return Parameters:

None.

Event(s) generated (unless masked away):

The Host Controller sends the Command Status event for this command to the Host when it has received the Sniff_Mode command. The Mode Change event will occur when the Sniff Mode has started for the specified connection handle. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, only the Mode Change event will indicate that this

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 328 of 1169

command has been completed. If an error occurs after the Command Status event has occurred, then the status in the Mode Change event will indicate the error.

11.2.6.3 Exit_Sniff_Mode

Command	OCF	Command parameters	Return parameters
HCI_Exit_Sniff_Mode	0x0004	Connection_Handle	—

Description:

The Exit_Sniff_Mode command is used to end the sniff mode for a connection handle, which is currently in sniff mode. The Link Manager will determine and issue the appropriate LMP commands to remove the sniff mode for the associated Connection Handle. Note that in addition, the connection handle cannot be one of SCO link type.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Return Parameters:

None.

Event(s) generated (unless masked away):

A Command Status event for this command will occur when Host Controller has received the Exit_Sniff_Mode command. The Mode Change event will occur when the Sniff Mode has ended for the specified connection handle. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, only the Mode Change event will indicate that this command has been completed.

11.2.6.4 Park_Mode

Command	OCF	Command parameters	Return parameters
HCI_Park_Mode	0x0005	Connection_Handle, Beacon_Max_Interval, Beacon_Min_Interval	—

Description:

The Park Mode command is used to alter the behavior of the Link Manager, and have the LM place the baseband connection associated by the specified Connection Handle into the Park mode. The Connection_Handle command parameter is used to identify which connection is to be placed in Park mode. The Connection_Handle shall be a Connection_Handle for an ACL connection. The Beacon Interval

Copyright © 2002 IEEE. All rights reserved.

command parameters specify the acceptable length of the interval between beacons. However, the remote device may request shorter interval. The Beacon_Max_Interval parameter specifies the acceptable longest length of the interval between beacons. The Beacon_Min_Interval parameter specifies the acceptable shortest length of the interval between beacons. Therefore, the Beacon Min Interval cannot be greater than the Beacon Max Interval. The Host Controller will return the actual Beacon interval in the Interval parameter of the Mode Change event, if the command is successful. This command enables the Host to support a low-power policy for itself or several other Bluetooth devices, allows the devices to enter Inquiry Scan, Page Scan, provides support for large number of Bluetooth Devices in a single piconet, and a number of other possible activities.

Note that when the Host issues the Park_Mode command, no Connection Handles for SCO connections are allowed to exist to the remote device that is identified by the Connection_Handle parameter. If one or more Connection Handles for SCO connections exist to that device, depending on the implementation, a Command Status event or a Mode Change event (following a Command Status event where Status=0x00) will be returned with the error code 0x0C “Command Disallowed”. If the Host sends data to the Host Controller with a Connection_Handle corresponding to a parked connection, the Host Controller will keep the data in its buffers until either the data can be transmitted (after unpark) or a flush, a flush timeout or a disconnection occurs. This is valid even if the Host has not yet been notified of the park mode through a Mode Change event when it sends the data.

Note that the above is not valid for an HCI Data Packet sent from the Host to the Host Controller on the master side where the Connection_Handle is a Connection_Handle used for Piconet Broadcast and the Broadcast_Flag is set to Piconet Broadcast. In that case, slaves in park mode will also receive the broadcast data. (If the Broadcast_Flag is set to Active Broadcast, the broadcast data will usually not be received by slaves in park mode.) It is possible for the Host Controller to do an automatic unpark to transmit data and then park the connection again depending on the value of the Link_Policy_Settings parameter (see Write_Link_Policy_Settings) and depending on whether the implementation supports this or not (optional feature). The optional feature of automatic unpark/park can also be used for link supervision. Whether Mode Change events are returned or not at automatic unpark/park if this is implemented, is vendor specific. This could be controlled by a vendor-specific HCI command.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Beacon_Max_Interval: *Size: 2 Bytes*

Value	Parameter description
N = 0xXXXX	Maximum acceptable number of Baseband slots between consecutive beacons. Interval Length = *N* * 0.625 ms (1 Baseband slot) Range for *N*: 0x0001–0xFFFF Time Range: 0.625 ms–40.9 s

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 330 of 1169

Beacon_Min_Interval *Size: 2 Bytes*

Value	Parameter description
N = 0xXXXX	Minimum acceptable number of Baseband slots between consecutive beacons Interval Length = *N* * 0.625 ms(1 Baseband slot) Range for *N*: 0x0001–0xFFFF Time Range: 0.625 ms–40.9 s

Return Parameters:

None.

Event(s) generated (unless masked away):

The Host Controller sends the Command Status event for this command to the Host when it has received the Park_Mode command. The Mode Change event will occur when the Park Mode has started for the specified connection handle. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, only the Mode Change event will indicate that this command has been completed. If an error occurs after the Command Status event has occurred, then the status in the Mode Change event will indicate the error.

11.2.6.5 Exit_Park_Mode

Command	OCF	Command parameters	Return parameters
HCI_Exit_Park_Mode	0x0006	Connection_Handle	—

Description:

The Exit_Park_Mode command is used to switch the Bluetooth device from park mode back to active mode. This command may only be issued when the device associated with the specified Connection Handle is in Park Mode. The Connection_Handle shall be a Connection_Handle for an ACL connection. This function does not complete immediately.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Return Parameters:

None.

Copyright © 2002 IEEE. All rights reserved.

Event(s) generated (unless masked away):

A Command Status event for this command will occur when the Host Controller has received the Exit_Park_Mode command. The Mode Change event will occur when the Park Mode has ended for the specified connection handle. Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, only the Mode Change event will indicate that this command has been completed.

11.2.6.6 QoS_Setup

Command	OCF	Command parameters	Return parameters
HCI_QoS_Setup	0x0007	Connection_Handle, Flags, Service_Type, Token_Rate, Peak_Bandwidth, Latency, Delay_Variation	—

Description:

The QoS_Setup command is used to specify Quality of Service parameters for a connection handle. The Connection_Handle shall be a Connection_Handle for an ACL connection. These QoS parameter are the same parameters as L2CAP QoS. For more detail see Clause 10, “Logical Link Control and Adaptation Protocol Specification”. This allows the Link Manager to have all of the information about what the Host is requesting for each connection. The LM will determine if the QoS parameters can be met. Bluetooth devices that are both slaves and masters can use this command. When a device is a slave, this command will trigger an LMP request to the master to provide the slave with the specified QoS as determined by the LM. When a device is a master, this command is used to request a slave device to accept the specified QoS as determined by the LM of the master. The Connection_Handle command parameter is used to identify for which connection the QoS request is requested.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify which connection for the QoS Setup. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Flags: *Size: 1 Byte*

Value	Parameter description
0x00–0xFF	Reserved for future use.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 332 of 1169

Service_Type: *Size: 1 Byte*

Value	Parameter description
0x00	No Traffic.
0x01	Best Effort.
0x02	Guaranteed.
0x03–0xFF	Reserved for future use.

Token_Rate: *Size: 4 Bytes*

Value	Parameter description
0xXXXXXXXX	Token Rate in bytes per second.

Peak_Bandwidth: *Size: 4 Bytes*

Value	Parameter description
0xXXXXXXXX	Peak Bandwidth in bytes per second.

Latency: *Size: 4 Bytes*

Value	Parameter description
0xXXXXXXXX	Latency in microseconds.

Delay_Variation: *Size: 4 Bytes*

Value	Parameter description
0xXXXXXXXX	Delay Variation in microseconds.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Host Controller receives the QoS_Setup command, the Host Controller sends the Command Status event to the Host. When the Link Manager has completed the LMP messages to establish the requested QoS parameters, the Host Controller on the local Bluetooth device will send a QoS Setup Complete event to the Host, and the event may also be generated on the remote side if there was LMP negotiation. The values of the parameters of the QoS Setup Complete event may, however, be different on the initiating and the remote side. The QoS Setup Complete event returned by the Host Controller on the local side contains the status of this command, and returned QoS parameters describing the supported QoS for the connection.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the QoS Setup Complete event will indicate that this command has been completed.

Copyright © 2002 IEEE. All rights reserved.

11.2.6.7 Role_Discovery

Command	OCF	Command parameters	Return parameters
HCI_Role_Discovery	0x0009	Connection_Handle	Status, Connection_Handle, Current_Role

Description:

The Role_Discovery command is used for a Bluetooth device to determine which role the device is performing for a particular Connection Handle. The Connection_Handle shall be a Connection_Handle for an ACL connection.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Role_Discovery command succeeded.
0x01–0xFF	Role_Discovery command failed. See Table 109 for list of Error Codes.

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify which connection the Role_Discovery command was issued on. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Current_Role: *Size: 1 Byte*

Value	Parameter description
0x00	Current Role is Master for this Connection Handle.
0x01	Current Role is Slave for this Connection Handle.

Event(s) generated (unless masked away):

When the Role_Discovery command has completed, a Command Complete event will be generated.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 334 of 1169

11.2.6.8 Switch_Role

Command	OCF	Command parameters	Return parameters
HCI_Switch_Role	0x000B	BD_ADDR, Role	—

Description:

The Switch_Role command is used for a Bluetooth device to switch the current role the device is performing for a particular connection with another specified Bluetooth device. The BD_ADDR command parameter indicates for which connection the role switch is to be performed. The Role indicates the requested new role that the local device performs. Note that the BD_ADDR command parameter shall specify a Bluetooth device for which a connection already exists.

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR for the connected device with which a role switch is to be performed.

Role: *Size: 1 Byte*

Value	Parameter description
0x00	Change own Role to Master for this BD_ADDR.
0x01	Change own Role to Slave for this BD_ADDR.

Return Parameters:

None.

Event(s) generated (unless masked away):

A Command Status event for this command will occur when the Host Controller has received the Switch_Role command. When the role switch is performed, a Role Change event will occur to indicate that the roles have been changed, and will be communicated to both Hosts.

Note that no Command Complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, only the Role Change event will indicate that this command has been completed.

Copyright © 2002 IEEE. All rights reserved.

11.2.6.9 Read_Link_Policy_Settings

Command	OCF	Command parameters	Return parameters
HCI_Read_Link_Policy_Settings	0x000C	Connection_Handle	Status, Connection_Handle Link_Policy_Settings

Description:

This command will read the Link Policy setting for the specified Connection Handle. The Link_Policy_Settings parameter determines the behavior of the local Link Manager when it receives a request from a remote device or it determines itself to change the master-slave role or to enter the hold, sniff, or park mode. The local Link Manager will automatically accept or reject such a request from the remote device, and may even autonomously request itself, depending on the value of the Link_Policy_Settings parameter for the corresponding Connection_Handle. When the value of the Link_Policy_Settings parameter is changed for a certain Connection_Handle, the new value will only be used for requests from a remote device or from the local Link Manager itself made after this command has been completed. The Connection_Handle shall be a Connection_Handle for an ACL connection. By enabling each mode individually, the Host can choose any combination needed to support various modes of operation. Multiple LM policies may be specified for the Link_Policy_Settings parameter by performing a bitwise OR operation of the different activity types.

Note that the local device may be forced into hold mode (regardless of whether the local device is master or slave) by the remote device regardless of the value of the Link_Policy_Settings parameter. The forcing of hold mode can, however, only be done once the connection has already been placed into hold mode through an LMP request (the Link_Policy_Settings determine if requests from a remote device should be accepted or rejected). The forcing of hold mode can after that be done as long as the connection lasts regardless of the setting for the hold mode in the Link_Policy_Settings parameter.

Note that the previous description implies that if the implementation in the remote device is a “polite” implementation that does not force another device into hold mode via LMP PDUs, then the Link_Policy_Settings will never be overruled.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Read_Link_Policy_Settings command succeeded.
0x01–0xFF	Read_Link_Policy_Settings command failed. See Table 109 for list of Error Codes.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 336 of 1169

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Link_Policy_Settings *Size: 2 Bytes*

Value	Parameter description
0x0000	Disable All LM Modes.
0x0001	Enable Master Slave Switch.
0x0002	Enable Hold Mode.
0x0004	Enable Sniff Mode.
0x0008	Enable Park Mode.
0x0010–0x8000	Reserved for future use.

Event(s) generated (unless masked away):

When the Read_Link_Policy_Settings command has completed, a Command Complete event will be generated.

11.2.6.10 Write_Link_Policy_Settings

Command	OCF	Command parameters	Return parameters
HCI_Write_Link_Policy_Settings	0x000D	Connection_Handle, Link_Policy_Settings	Status, Connection_Handle

Description:

This command will write the Link Policy setting for the specified Connection Handle. The Link_Policy_Settings parameter determines the behavior of the local Link Manager when it receives a request from a remote device or it determines itself to change the master-slave role or to enter the hold, sniff, or park mode. The local Link Manager will automatically accept or reject such a request from the remote device, and may even autonomously request itself, depending on the value of the Link_Policy_Settings parameter for the corresponding Connection_Handle. When the value of the Link_Policy_Settings parameter is changed for a certain Connection_Handle, the new value will only be used for requests from a remote device or from the local Link Manager itself made after this command has been completed. The Connection_Handle shall be a Connection_Handle for an ACL connection. By enabling each mode individually, the Host can choose any combination needed to support various modes of operation. Multiple LM policies may be specified for the Link_Policy_Settings parameter by performing a bitwise OR operation of the different activity types.

Note that the local device may be forced into hold mode (regardless of whether the local device is master or slave) by the remote device regardless of the value of the Link_Policy_Settings parameter. The forcing of hold mode can however only be done once the connection has already been placed into hold mode through

Copyright © 2002 IEEE. All rights reserved.

an LMP request (the Link_Policy_Settings determine if requests from a remote device should be accepted or rejected). The forcing of hold mode can after that be done as long as the connection lasts regardless of the setting for the hold mode in the Link_Policy_Settings parameter.

Note that the previous description implies that if the implementation in the remote device is a "polite" implementation that does not force another device into hold mode via LMP PDUs, then the Link_Policy_Settings will never be overruled.

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use)

Link_Policy_Settings *Size: 2 Bytes*

Value	Parameter description
0x0000	Disable All LM Modes **Default**.
0x0001	Enable Master Slave Switch.
0x0002	Enable Hold Mode.
0x0004	Enable Sniff Mode.
0x0008	Enable Park Mode.
0x0010–0x8000	Reserved for future use.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Write_Link_Policy_Settings command succeeded.
0x01–0xFF	Write_Link_Policy_Settings command failed. See Table 109 for list of Error Codes.

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify a connection. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use)

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 338 of 1169

Event(s) generated (unless masked away):

When the Write_Link_Policy_Settings command has completed, a Command Complete event will be generated.

11.2.7 Host controller and baseband commands

The Host Controller and Baseband Commands provide access and control to various capabilities of the Bluetooth hardware. These parameters provide control of Bluetooth devices and of the capabilities of the Host Controller, Link Manager, and Baseband. The host device can use these commands to modify the behavior of the local device. For the HCI Control and Baseband Commands, the OGF is defined as 0x03.

Command	Command summary description
Set_Event_Mask	The Set_Event_Mask command is used to control which events are generated by the HCI for the Host.
Reset	The Reset command will reset the Bluetooth Host Controller, Link Manager, and the radio module.
Set_Event_Filter	The Set_Event_Filter command is used by the Host to specify different event filters. The Host may issue this command multiple times to request various conditions for the same type of event filter and for different types of event filters.
Flush	The Flush command is used to discard all data that is currently pending for transmission in the Host Controller for the specified connection handle, even if there currently are chunks of data that belong to more than one L2CAP packet in the Host Controller.
Read_PIN_Type	The Read_PIN_Type command is used for the Host to read the value that is specified to indicate whether the Host supports variable PIN or only fixed PINs.
Write_PIN_Type	The Write_PIN_Type command is used for the Host to specify whether the Host supports variable PIN or only fixed PINs.
Create_New_Unit_Key	The Create_New_Unit_Key command is used to create a new unit key.
Read_Stored_Link_Key	The Read_Stored_Link_Key command provides the ability to read one or more link keys stored in the Bluetooth Host Controller.
Write_Stored_Link_Key	The Write_Stored_Link_Key command provides the ability to write one or more link keys to be stored in the Bluetooth Host Controller.

Copyright © 2002 IEEE. All rights reserved.

Command	Command summary description
Delete_Stored_Link_Key	The Delete_Stored_Link_Key command provides the ability to remove one or more of the link keys stored in the Bluetooth Host Controller.
Change_Local_Name	The Change_Local_Name command provides the ability to modify the user-friendly name for the Bluetooth device.
Read_Local_Name	The Read_Local_Name command provides the ability to read the stored user-friendly name for the Bluetooth device.
Read_Connection_Accept_Timeout	The Read_Connection_Accept_Timeout command will read the value for the Connection_Accept_Timeout configuration parameter, which allows the Bluetooth hardware to automatically deny a connection request after a specified period has occurred, and to refuse a new connection.
Write_Connection_Accept_Timeout	The Write_Connection_Accept_Timeout will write the value for the Connection_Accept_ Timeout configuration parameter, which allows the Bluetooth hardware to automatically deny a connection request after a specified period has occurred, and to refuse a new connection.
Read_Page_Timeout	The Read_Page_Timeout command will read the value for the Page_Reply_Timeout configuration parameter, which allows the Bluetooth hardware to define the amount of time a connection request will wait for the remote device to respond before the local device returns a connection failure.
Write_Page_Timeout	The Write_Page_Timeout command will write the value for the Page_Reply_Timeout configuration parameter, which allows the Bluetooth hardware to define the amount of time a connection request will wait for the remote device to respond before the local device returns a connection failure.
Read_Scan_Enable	The Read_Scan_Enable command will read the value for the Scan_Enable configuration parameter, which controls whether or not the Bluetooth device will periodically scan for page attempts and/or inquiry requests from other Bluetooth devices.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 340 of 1169

Command	Command summary description
Write_Scan_Enable	The Write_Scan_Enable command will write the value for the Scan_Enable configuration parameter, which controls whether or not the Bluetooth device will periodically scan for page attempts and/or inquiry requests from other Bluetooth devices.
Read_Page_Scan_Activity	The Read_Page_Scan_Activity command will read the values for the Page_Scan_Interval and Page_Scan_Window configuration parameters. Page_Scan_Interval defines the amount of time between consecutive page scans. Page_Scan_Window defines the duration of the page scan.
Write_Page_Scan_Activity	The Write_Page_Scan_Activity command will write the value for Page_Scan_Interval and_ Page_Scan_Window configuration parameters. Page_Scan_Interval defines the amount of time between consecutive page scans. Page_Scan_Window defines the duration of the page scan.
Read_Inquiry_Scan_Activity	The Read_Inquiry_Scan_Activity command will read the value for Inquiry_Scan_Interval and Inquiry_Scan_Window configuration parameters. Inquiry_Scan_Interval defines the amount of time between consecutive inquiry scans. Inquiry_Scan_Window defines the amount of time for the duration of the inquiry scan.
Write_Inquiry_Scan_Activity	The Write_Inquiry_Scan_Activity command will write the value for Inquiry_Scan_Interval and Inquiry_Scan_Window configuration parameters. Inquiry_Scan_Interval defines the amount of time between consecutive inquiry scans. Inquiry_Scan_Window defines the amount of time for the duration of the inquiry scan.
Read_Authentication_Enable	The Read_Authentication_Enable command will read the value for the Authentication_ Enable parameter, which controls whether the Bluetooth device will require authentication for each connection with other Bluetooth devices.
Write_Authentication_Enable	The Write_Authentication_Enable command will write the value for the Authentication_ Enable parameter, which controls whether the Bluetooth device will require authentication for each connection with other Bluetooth devices.

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 341 of 1169

Command	Command summary description
Read_Encryption_Mode	The Read_Encryption_Mode command will read the value for the Encryption_Mode parameter, which controls whether the Bluetooth device will require encryption for each connection with other Bluetooth devices.
Write_Encryption_Mode	The Write_Encryption_Mode command will write the value for the Encryption_Mode parameter, which controls whether the Bluetooth device will require encryption for each connection with other Bluetooth devices.
Read_Class_of_Device	The Read_Class_of_Device command will read the value for the Class_of_Device parameter, which is used to indicate its capabilities to other devices.
Write_Class_of_Device	The Write_Class_of_Device command will write the value for the Class_of_Device parameter, which is used to indicate its capabilities to other devices.
Read_Voice_Setting	The Read_Voice_Setting command will read the values for the Voice_Setting parameter, which controls all the various settings for the voice connections.
Write_Voice_Setting	The Write_Voice_Setting command will write the values for the Voice_Setting parameter, which controls all the various settings for the voice connections.
Read_Automatic_Flush_Timeout	The Read_Automatic_Flush_Timeout will read the value for the Flush_Timeout parameter for the specified connection handle. The Flush_ Timeout parameter is only used for ACL connections.
Write_Automatic_Flush_Timeout	The Write_Automatic_Flush_Timeout will write the value for the Flush_Timeout parameter for the specified connection handle. The Flush_ Timeout parameter is only used for ACL connections.
Read_Num_Broadcast_Retransmissions	The Read_Num_Broadcast_Retransmissions command will read the parameter value for the Number of Broadcast Retransmissions for the device. Broadcast packets are not acknowledged and are unreliable. This parameter is used to increase the reliability of a broadcast message by retransmitting the broadcast message multiple times.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 342 of 1169

Command	Command summary description
Write_Num_Broadcast_Retransmissions	The Write_Num_Broadcast_Retransmissions command will write the parameter value for the Number of Broadcast Retransmissions for the device. Broadcast packets are not acknowledged and are unreliable. This parameter is used to increase the reliability of a broadcast message by retransmitting the broadcast message multiple times.
Read_Hold_Mode_Activity	The Read_Hold_Mode_Activity command will read the value for the Hold_Mode_Activity parameter. This value is used to determine what activity the device should do when it is in hold mode.
Write_Hold_Mode_Activity	The Write_Hold_Mode_Activity command will write the value for the Hold_Mode_Activity parameter. This value is used to determine what activity the device should do when it is in hold mode.
Read_Transmit_Power_Level	The Read_Transmit_Power_Level command will read the values for the Transmit_Power_ Level parameter for the specified Connection Handle.
Read_SCO_Flow_Control_Enable	The Read_SCO_Flow_Control_Enable command provides the ability to read the SCO_Flow_Control_Enable setting. By using this setting, the Host can decide if the Host Controller will send Number Of Completed Packets events for SCO Connection Handles.
Write_SCO_Flow_Control_Enable	The Write_SCO_Flow_Control_Enable command provides the ability to write the SCO_Flow_Control_Enable setting. By using this setting, the Host can decide if the Host Controller will send Number Of Completed Packets events for SCO Connection Handles.
Read_Link_Supervision_Timeout	The Read_Link_Supervision_Timeout command will read the value for the Link_Supervision_Timeout parameter for the device. This parameter is used by the master or slave Bluetooth device to monitor link loss.
Write_Link_Supervision_Timeout	The Write_Link_Supervision_Timeout command will write the value for the Link_Supervision_ Timeout parameter for the device. This parameter is used by the master or slave Bluetooth device to monitor link loss.

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 343 of 1169

Command	Command summary description
Read_Number_Of_Supported_IAC	The Read_Number_Of_Supported_IAC command will read the value for the number of Inquiry Access Codes (IAC) that the local Bluetooth device can simultaneously listen for during an Inquiry Scan.
Read_Current_IAC_LAP	The Read_Current_IAC_LAP command will read the LAP(s) used to create the Inquiry Access Codes (IAC) that the local Bluetooth device is simultaneously scanning for during Inquiry Scans.
Write_Current_IAC_LAP	The Write_Current_IAC_LAP will write the LAP(s) used to create the Inquiry Access Codes (IAC) that the local Bluetooth device is simultaneously scanning for during Inquiry Scans.
Read_Page_Scan_Period_Mode	The Read_Page_Scan_Period_Mode command is used to read the mandatory Page_Scan_Period_Mode of the local Bluetooth device.
Write_Page_Scan_Period_Mode	The Write_Page_Scan_Period_Mode command is used to write the mandatory Page_Scan_Period_Mode of the local Bluetooth device.
Read_Page_Scan_Mode	The Read_Page_Scan_Mode command is used to read the default Page_Scan_Mode of the local Bluetooth device.
Write_Page_Scan_Mode	The Write_Page_Scan_Mode command is used to write the default Page_Scan_Mode of the local Bluetooth device.

11.2.7.1 Set_Event_Mask

Command	OCF	Command parameters	Return parameters
HCI_Set_Event_Mask	0x0001	Event_Mask	Status

Description:

The Set_Event_Mask command is used to control which events are generated by the HCI for the Host. If the bit in the Event_Mask is set to a one, then the event associated with that bit will be enabled. The Host has to deal with each event that occurs by the Bluetooth devices. The event mask allows the Host to control how much it is interrupted.

Note that the Command Complete event, Command Status event and Number Of Completed Packets event cannot be masked. These events always occur. The Event_Mask is a bit mask of all of the events specified in Table 108.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 344 of 1169

Command Parameters:

Event_Mask: *Size: 8 Bytes*

Value	Parameter description
0x0000000000000000	No events specified.
0x0000000000000001	Inquiry Complete event.
0x0000000000000002	Inquiry Result event.
0x0000000000000004	Connection Complete event.
0x0000000000000008	Connection Request event.
0x0000000000000010	Disconnection Complete event.
0x0000000000000020	Authentication Complete event.
0x0000000000000040	Remote Name Request Complete event.
0x0000000000000080	Encryption Change event.
0x0000000000000100	Change Connection Link Key Complete event.
0x0000000000000200	Master Link Key Complete event.
0x0000000000000400	Read Remote Supported Features Complete event.
0x0000000000000800	Read Remote Version Information Complete event.
0x0000000000001000	QoS Setup Complete event.
0x0000000000002000	Command Complete event.
0x0000000000004000	Command Status event.
0x0000000000008000	Hardware Error event.
0x0000000000010000	Flush Occurred event.
0x0000000000020000	Role Change event.

Value	Parameter description
0x0000000000040000	Number Of Completed Packets event.
0x0000000000080000	Mode Change event.
0x0000000000100000	Return Link Keys event.
0x0000000000200000	PIN Code Request event.
0x0000000000400000	Link Key Request event.
0x0000000000800000	Link Key Notification event.
0x0000000001000000	Loopback Command event.

Copyright © 2002 IEEE. All rights reserved.

Value	Parameter description
0x0000000002000000	Data Buffer Overflow event.
0x0000000004000000	Max Slots Change event.
0x0000000008000000	Read Clock Offset Complete event.
0x0000000010000000	Connection Packet Type Changed even.t
0x0000000020000000	QoS Violation event.
0x0000000040000000	Page Scan Mode Change event .
0x0000000080000000	Page Scan Repetition Mode Change event.
0x0000000100000000 to 0x8000000000000000	Reserved for future use.
0x00000000FFFFFFFF	**Default** (All events enabled).

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Set_Event_Mask command succeeded.
0x01-0xFF	Set_Event_Mask command failed. See Table 109 for list of Error Codes.

Event(s) generated (unless masked away):

When the Set_Event_Mask command has completed, a Command Complete event will be generated.

11.2.7.2 Reset

Command	OCF	Command parameters	Return parameters
HCI_Reset	0x0003	—	Status

Description:

The Reset command will reset the Host Controller and the Link Manager. The reset command should not affect the used HCI transport layer since the HCI transport layers have reset mechanisms of their own. After the reset is completed, the current operational state will be lost, the Bluetooth device will enter standby mode and the Host Controller will automatically revert to the default values for the parameters for which default values are defined in the specification.

Note that the Host is not allowed to send additional HCI commands before the Command Complete event related to the Reset command has been received.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 346 of 1169

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Reset command succeeded, was received and will be executed.
0x01–0xFF	Reset command failed. See Table 109 for list of Error Codes.

Event(s) generated (unless masked away):

When the reset has been performed, a Command Complete event will be generated.

11.2.7.3 Set_Event_Filter

Command	OCF	Command parameters	Return parameters
HCI_Set_Event_Filter	0x0005	Filter_Type, Filter_Condition_Type, Condition	Status

Description:

The Set_Event_Filter command is used by the Host to specify different event filters. The Host may issue this command multiple times to request various conditions for the same type of event filter and for different types of event filters. The event filters are used by the Host to specify items of interest, which allow the Host Controller to send only events that interest the Host. Only some of the events have event filters. By default (before this command has been issued after power-on or Reset), no filters are set, and the Auto_Accept_Flag is off (incoming connections are not automatically accepted). An event filter is added each time this command is sent from the Host and the Filter_Condition_Type is not equal to 0x00. (The old event filters will not be overwritten). To clear all event filters, the Filter_Type = 0x00 is used. The Auto_Accept_Flag will then be set to off. To clear event filters for only a certain Filter_Type, the Filter_Condition_Type = 0x00 is used. The Inquiry Result filter allows the Host Controller to filter out Inquiry Result events. The Inquiry Result filter allows the Host to specify that the Host Controller only sends Inquiry Results to the Host if the Inquiry Result event meets one of the specified conditions set by the Host. For the Inquiry Result filter, the Host can specify one or more of the following Filter Condition Types:

1) A new device responded to the Inquiry process
2) A device with a specific Class of Device responded to the Inquiry process
3) A device with a specific BD_ADDR responded to the Inquiry process

The Inquiry Result filter is used in conjunction with the Inquiry and Periodic Inquiry command. The Connection Setup filter allows the Host to specify that the Host Controller only sends a Connection Complete or Connection Request event to the Host if the event meets one of the specified conditions set by the Host. For the Connection Setup filter, the Host can specify one or more of the following Filter Condition Types:

Copyright © 2002 IEEE. All rights reserved.

1) Allow Connections from all devices
2) Allow Connections from a device with a specific Class of Device
3) Allow Connections from a device with a specific BD_ADDR

For each of these conditions, an Auto_Accept_Flag parameter allows the Host to specify what action should be done when the condition is met. The Auto_Accept_Flag allows the Host to specify if the incoming connection should be auto accepted (in which case the Host Controller will send the Connection Complete event to the Host when the connection is completed) or if the Host should make the decision (in which case the Host Controller will send the Connection Request event to the Host, to elicit a decision on the connection).

The Connection Setup filter is used in conjunction with the Read/Write_Scan_Enable commands. If the local device is in the process of a page scan, and is paged by another device which meets one on the conditions set by the Host, and the Auto_Accept_Flag is off for this device, then a Connection Request event will be sent to the Host by the Host Controller. A Connection Complete event will be sent later on after the Host has responded to the incoming connection attempt. In this same example, if the Auto_Accept_Flag is on, then a Connection Complete event will be sent to the Host by the Host Controller. (No Connection Request event will be sent in that case.)

The Host Controller will store these filters in volatile memory until the Host clears the event filters using the Set_Event_Filter command or until the Reset command is issued. The number of event filters the Host Controller can store is implementation dependent. If the Host tries to set more filters than the Host Controller can store, the Host Controller will return the "Memory Full" error code and the filter will not be installed.

Note that the Clear All Filters has no Filter Condition Types or Conditions.

Note that in the condition that a connection is auto accepted, a Link Key Request event and possibly also a PIN Code Request event and a Link Key Notification event could be sent to the Host by the Host Controller before the Connection Complete event is sent.

If there is a contradiction between event filters, the latest set event filter will override older ones. An example is an incoming connection attempt where more than one Connection Setup filter matches the incoming connection attempt, but the Auto-Accept_Flag has different values in the different filters.

Command Parameters:

Filter_Type: *Size: 1 Byte*

Value	Parameter description
0x00	Clear All Filters (Note—In this case, the Filter_Condition_type and Condition parameters should not be given, they should have a length of 0 bytes. Filter_Type should be the only parameter.)
0x01	Inquiry Result.
0x02	Connection Setup.
0x03–0xFF	Reserved for future use.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 348 of 1169

Filter Condition Types: For each Filter Type one or more Filter Condition types exists.

Inquiry_Result_Filter_Condition_Type: *Size: 1 Byte*

Value	Parameter description
0x00	A new device responded to the Inquiry process. (Note that a device may be reported to the Host in an Inquiry Result event more than once during an inquiry or inquiry period depending on the implementation. See description in 11.2.5.1 and 11.2.5.3.)
0x01	A device with a specific Class of Device responded to the Inquiry process.
0x02	A device with a specific BD_ADDR responded to the Inquiry process.
0x03–0xFF	Reserved for future use.

Connection_Setup_Filter_Condition_Type: *Size: 1 Byte*

Value	Parameter description
0x00	Allow Connections from all devices.
0x01	Allow Connections from a device with a specific Class of Device.
0x02	Allow Connections from a device with a specific BD_ADDR.
0x03–0xFF	Reserved for future use.

Condition: For each Filter Condition Type defined for the Inquiry Result Filter and the Connection Setup Filter, zero or more Condition parameters are required, depending on the filter condition type and filter type.

Condition for Inquiry_Result_Filter_Condition_Type = 0x00

Condition: *Size: 0 Byte*

Value	Parameter description
	The Condition parameter is not used.

Condition for Inquiry_Result_Filter_Condition_Type = 0x01

Condition:

Size: 6 Bytes

Class_of_Device: *Size: 3 Bytes*

Value	Parameter description
0x000000	Default, Return All Devices.
0xXXXXXX	*Class of Device* of Interest.

Copyright © 2002 IEEE. All rights reserved.

Class_of_Device_Mask: *Size: 3 Bytes*

Value	Parameter description
0xXXXXXX	Bit Mask used to determine which bits of the Class of Device parameter are “don’t care”. Zero-value bits in the mask indicate the “don’t care” bits of the Class of Device.

Condition for Inquiry_Result_Filter_Condition_Type = 0x02

Condition:

Size: 6 Bytes

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device of Interest.

Condition for Connection_Setup_Filter_Condition_Type = 0x00

Condition:

Size: 1 Byte

Auto_Accept_Flag: *Size:1 Byte*

Value	Parameter description
0x01	Do NOT Auto accept the connection. (Auto accept is off.)
0x02	Do Auto accept the connection with role switch disabled. (Auto accept is on.)
0x03	Do Auto accept the connection with role switch enabled. (Auto accept is on). Note: When auto accepting an incoming SCO connection, no role switch will be performed. The value 0x03 of the Auto_Accept_Flag will then get the same effect as if the value had been 0x02.
0x04–0xFF	Reserved for future use.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 350 of 1169

Condition for Connection_Setup_Filter_Condition_Type = 0x01

Condition:

Size: 7 Bytes

Class_of_Device: *Size: 3 Bytes*

Value	Parameter description
0x000000	Default, Return All Devices.
0xXXXXXX	*Class of Device* of Interest.

Class_of_Device_Mask: *Size: 3 Bytes*

Value	Parameter description
0xXXXXXX	Bit Mask used to determine which bits of the Class of Device parameter are “don’t care”. Zero-value bits in the mask indicate the “don’t care” bits of the Class of Device.

Auto_Accept_Flag: *Size: 1 Byte*

Value	Parameter description
0x01	Do NOT Auto accept the connection. (Auto accept is off.)
0x02	Do Auto accept the connection with role switch disabled. (Auto accept is on.)
0x03	Do Auto accept the connection with role switch enabled. (Auto accept is on). Note—When auto accepting an incoming SCO connection, no role switch will be performed. The value 0x03 of the Auto_Accept_Flag will then get the same effect as if the value had been 0x02.
0x04–0xFF	Reserved for future use.

Condition for Connection_Setup_Filter_Condition_Type = 0x02

Condition:

Size: 7 Bytes

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR of the Device of Interest.

Copyright © 2002 IEEE. All rights reserved.

Auto_Accept_Flag: *Size: 1 Byte*

Value	Parameter description
0x01	Do NOT Auto accept the connection. (Auto accept is off.)
0x02	Do Auto accept the connection with role switch disabled. (Auto accept is on.)
0x03	Do Auto accept the connection with role switch enabled. (Auto accept is on). Note: When auto accepting an incoming SCO connection, no role switch will be performed. The value 0x03 of the Auto_Accept_Flag will then get the same effect as if the value had been 0x02.
0x04–0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Set_Event_Filter command succeeded.
0x01–0xFF	Set_Event_Filter command failed. See Table 109 for list of Error Codes.

Event(s) generated (unless masked away):

A Command Complete event for this command will occur when the Host Controller has enabled the filtering of events. When one of the conditions are met, a specific event will occur.

11.2.7.4 Flush

Command	OCF	Command parameters	Return parameters
HCI_Flush	0x0008	Connection_Handle	Status, Connection_Handle

Description:

The Flush command is used to discard all data that is currently pending for transmission in the Host Controller for the specified connection handle, even if there currently are chunks of data that belong to more than one L2CAP packet in the Host Controller. After this, all data that is sent to the Host Controller for the same connection handle will be discarded by the Host Controller until an HCI Data Packet with the start Packet_Boundary_Flag (0x02) is received. When this happens, a new transmission attempt can be made. This command will allow higher-level software to control how long the baseband should try to retransmit a baseband packet for a connection handle before all data that is currently pending for transmission in the Host Controller should be flushed. Note that the Flush command is used for ACL connections ONLY. In addition to the Flush command, the automatic flush timers (see 11.2.7.31) can be used to automatically flush the L2CAP packet that is currently being transmitted after the specified flush timer has expired.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 352 of 1169

Command Parameters:

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify which connection to flush. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Flush command succeeded.
0x01–0xFF	Flush command failed. See Table 109 for list of Error Codes.

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

Value	Parameter description
0xXXXX	Connection Handle to be used to identify which connection the flush command was issued on. Range: 0x0000–0x0EFF (0x0F00–0x0FFF Reserved for future use).

Event(s) generated (unless masked away):

The Flush Occurred event will occur once the flush is completed. A Flush Occurred event could be from an automatic Flush or could be caused by the Host issuing the Flush command. When the Flush command has completed, a Command Complete event will be generated, to indicate that the Host caused the Flush.

11.2.7.5 Read_PIN_Type

Command	OCF	Command parameters	Return parameters
HCI_Read_PIN_Type	0x0009	—	Status, PIN_Type

Description:

The Read_PIN_Type command is used for the Host to read whether the Link Manager assumes that the Host supports variable PIN codes only a fixed PIN code. The Bluetooth hardware uses the PIN-type information during pairing.

Command Parameters:

None.

Copyright © 2002 IEEE. All rights reserved.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Read_PIN_Type command succeeded.
0x01-0xFF	Read_PIN_Type command failed. See Table 109 for list of Error Codes.

PIN_Type: *Size: 1 Byte*

Value	Parameter description
0x00	Variable PIN.
0x01	Fixed PIN.

Event(s) generated (unless masked away):

When the Read_PIN_Type command has completed, a Command Complete event will be generated.

11.2.7.6 Write_PIN_Type

Command	OCF	Command parameters	Return parameters
HCI_Write_PIN_Type	0x000A	PIN_Type	Status

Description:

The Write_PIN_Type command is used for the Host to write to the Host Controller whether the Host supports variable PIN codes or only a fixed PIN code. The Bluetooth hardware uses the PIN-type information during pairing.

Command Parameters:

PIN_Type: *Size: 1 Byte*

Value	Parameter description
0x00	Variable PIN.
0x01	Fixed PIN.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Write PIN Type command succeeded.
0x01–0xFF	Write PIN Type command failed. See Table 109 for list of Error Codes.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 354 of 1169

Event(s) generated (unless masked away):

When the Write_PIN_Type command has completed, a Command Complete event will be generated.

11.2.7.7 Create_New_Unit_Key

Command	OCF	Command parameters	Return parameters
HCI_Create_New_Unit_Key	0x000B	—	Status

Description:

The Create_New_Unit_Key command is used to create a new unit key. The Bluetooth hardware will generate a random seed that will be used to generate the new unit key. All new connection will use the new unit key, but the old unit key will still be used for all current connections.

Note that this command will not have any effect for a device that does not use unit keys (i.e., a device that uses only combination keys).

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Create New Unit Key command succeeded.
0x01–0xFF	Create New Unit Key command failed. See Table 109 for list of Error Codes.

Event(s) generated (unless masked away):

When the Create_New_Unit_Key command has completed, a Command Complete event will be generated.

11.2.7.8 Read_Stored_Link_Key

Command	OCF	Command parameters	Return parameters
HCI_Read_Stored_ Link_Key	0x000D	BD_ADDR, Read_All_Flag	Status, Max_Num_Keys, Num_Keys_Read

Description:

The Read_Stored_Link_Key command provides the ability to read one or more link keys stored in the Bluetooth Host Controller. The Bluetooth Host Controller can store a limited number of link keys for other Bluetooth devices. Link keys are shared between two Bluetooth devices, and are used for all security transactions between the two devices. A Host device may have additional storage capabilities, which can be

Copyright © 2002 IEEE. All rights reserved.

used to save additional link keys to be reloaded to the Bluetooth Host Controller when needed. The Read_All_Flag parameter is used to indicate if all of the stored Link Keys should be returned. If Read_All_Flag indicates that all Link Keys are to be returned, then the BD_ADDR command parameter shall be ignored The BD_ADDR command parameter is used to identify which link key to read. The stored Link Keys are returned by one or more Return Link Keys events.

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR for the stored link key to be read.

Read_All_Flag: *Size: 1 Byte*

Value	Parameter description
0x00	Return Link Key for specified BD_ADDR.
0x01	Return all stored Link Keys.
0x02–0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Read_Stored_Link_Key command succeeded.
0x01–0xFF	Read_Stored_Link_Key command failed. See Table 109 for list of Error Codes.

Max_Num_Keys: *Size: 2 Byte*

Value	Parameter description
0xXXXX	Total Number of Link Keys that the Host Controller can store. Range: 0x0000–0xFFFF

Num_Keys_Read: *Size: 2 Bytes*

Value	Parameter description
0xXXXX	Number of Link Keys Read. Range: 0x0000–0xFFFF

Event(s) generated (unless masked away):

Zero or more instances of the Return Link Keys event will occur after the command is issued. When there are no link keys stored, no Return Link Keys events will be returned. When there are link keys stored, the

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 356 of 1169

number of link keys returned in each Return Link Keys event is implementation specific. When the Read Stored Link Key command has completed a Command Complete event will be generated.

11.2.7.9 Write_Stored_Link_Key

Command	OCF	Command parameters	Return parameters
HCI_Write_Stored_Link_Key	0x0011	Num_Keys_To_Write, *BD_ADDR*[i], Link_Key[i]	Status, Num_Keys_Written

Description:

The Write_Stored_Link_Key command provides the ability to write one or more link keys to be stored in the Bluetooth Host Controller. The Bluetooth Host Controller can store a limited number of link keys for other Bluetooth devices. If no additional space is available in the Bluetooth Host Controller then no additional link keys will be stored. If space is limited and if all the link keys to be stored will not fit in the limited space, then the order of the list of link keys without any error will determine which link keys are stored. Link keys at the beginning of the list will be stored first. The Num_Keys_Written parameter will return the number of link keys that were successfully stored. If no additional space is available, then the Host must delete one or more stored link keys before any additional link keys are stored. The link key replacement algorithm is implemented by the Host and not the Host Controller. Link keys are shared between two Bluetooth devices and are used for all security transactions between the two devices. A Host device may have additional storage capabilities, which can be used to save additional link keys to be reloaded to the Bluetooth Host Controller when needed.

Note that Link Keys are only stored by issuing this command.

Command Parameters:

Num_Keys_To_Write: *Size: 1 Byte*

Value	Parameter description
0xXX	Number of Link Keys to Write. Range: 0x01–0x0B

BD_ADDR [i]: *Size: 6 Bytes * Num_Keys_To_Write*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR for the associated Link Key.

Link_Key[i]: *Size: 16 Bytes * Num_Keys_To_Write*

Value	Parameter description
0xXXXXXXXXXX XXXXXXXXXX XXXXXXXXXX	Link Key for the associated BD_ADDR.

Copyright © 2002 IEEE. All rights reserved.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Write_Stored_Link_Key command succeeded.
0x01–0xFF	Write_Stored_Link_Key command failed. See Table 109 for list of Error Codes.

Num_Keys_Written: *Size: 1 Bytes*

Value	Parameter description
0xXX	Number of Link Keys successfully written. Range: 0x00–0x0B

Event(s) generated (unless masked away):

When the Write_Stored_Link_Key command has completed, a Command Complete event will be generated.

11.2.7.10 Delete_Stored_Link_Key

Command	OCF	Command parameters	Return parameters
HCI_Delete_Stored_ Link_Key	0x0012	BD_ADDR, Delete_All_Flag	Status, Num_Keys_Deleted

Description:

The Delete_Stored_Link_Key command provides the ability to remove one or more of the link keys stored in the Bluetooth Host Controller. The Bluetooth Host Controller can store a limited number of link keys for other Bluetooth devices. Link keys are shared between two Bluetooth devices and are used for all security transactions between the two devices. The Delete_All_Flag parameter is used to indicate if all of the stored Link Keys should be deleted. If the Delete_All_Flag indicates that all Link Keys are to be deleted, then the BD_ADDR command parameter shall be ignored. This command provides the ability to negate all security agreements between two devices. The BD_ADDR command parameter is used to identify which link key to delete. If a link key is currently in use for a connection, then the link key will be deleted when all of the connections are disconnected.

Command Parameters:

BD_ADDR: *Size: 6 Bytes*

Value	Parameter description
0xXXXXXXXXXXXX	BD_ADDR for the link key to be deleted.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 358 of 1169

Delete_All_Flag: *Size: 1 Byte*

Value	Parameter description
0x00	Delete only the Link Key for specified BD_ADDR.
0x01	Delete all stored Link Keys.
0x02–0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Delete_Stored_Link_Key command succeeded.
0x01–0xFF	Delete_Stored_Link_Key command failed. See Table 109 for list of Error Codes.

Num_Keys_Deleted: *Size: 2 Bytes*

Value	Parameter description
0xXXXX	Number of Link Keys Deleted.

Event(s) generated (unless masked away):

When the Delete_Stored_Link_Key command has completed, a Command Complete event will be generated.

11.2.7.11 Change_Local_Name

Command	OCF	Command parameters	Return parameters
HCI_Change_Local_Name	0x0013	Name	Status

Description:

The Change_Local_Name command provides the ability to modify the user-friendly name for the Bluetooth device. A Bluetooth device may send a request to get the user-friendly name of another Bluetooth device. The user-friendly name provides the user with the ability to distinguish one Bluetooth device from another. The Name command parameter is a UTF-8 encoded string with up to 248 bytes in length. The Name command parameter should be null-terminated (0x00) if the UTF-8 encoded string is less than 248 bytes.

Note that the Name Parameter is a string parameter. Endianess does therefore not apply to the Name Parameter. The first byte of the name should be transmitted first.

Copyright © 2002 IEEE. All rights reserved.

Command Parameters:

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.
	Null terminated Zero length String. **Default.**

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Change_Local_Name command succeeded.
0x01–0xFF	Change_Local_Name command failed. See Table 109 for list of Error Codes.

Event(s) generated (unless masked away):

When the Change_Local_Name command has completed, a Command Complete event will be generated.

11.2.7.12 Read_Local_Name

Command	OCF	Command parameters	Return parameters
HCI_Read_Local_Name	0x0014	—	Status, Name

Description:

The Read_Local_Name command provides the ability to read the stored user-friendly name for the Bluetooth device. The user-friendly name provides the user the ability to distinguish one Bluetooth device from another. The Name return parameter is a UTF-8 encoded string with up to 248 bytes in length. The Name return parameter will be null terminated (0x00) if the UTF-8 encoded string is less than 248 bytes.

Note that the Name Parameter is a string parameter. Endianess does therefore not apply to the Name Parameter. The first byte of the name is received first.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 360 of 1169

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Read_Local_Name command succeeded.
0x01–0xFF	Read_Local_Name command failed. See Table 109 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.

Event(s) generated (unless masked away):

When the Read_Local_Name command has completed a Command Complete event will be generated.

11.2.7.13 Read_Connection_Accept_Timeout

Command	OCF	Command parameters	Return parameters
HCI_Read_Connection_Accept_Timeout	0x0015	—	Status, Conn_Accept_Timeout

Description:

This command will read the value for the Connection_Accept_Timeout configuration parameter. The Connection_Accept_Timeout configuration parameter allows the Bluetooth hardware to automatically deny a connection request after a specified time period has occurred and the new connection is not accepted. The parameter defines the time duration from when the Host Controller sends a Connection Request event until the Host Controller will automatically reject an incoming connection.

Command Parameters:

None.

Copyright © 2002 IEEE. All rights reserved.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Read_Connection_Accept_Timeout command succeeded.
0x01–0xFF	Read_Connection_Accept_Timeout command failed. See Table 109 for list of Error Codes.

Conn_Accept_Timeout: *Size: 2 Bytes*

Value	Parameter description
N = 0xXXXX	Connection Accept Timeout measured in Number of Baseband slots. Interval Length = *N* * 0.625 ms (1 Baseband slot) Range for *N*: 0x0001–0xB540 Time Range: 0.625 ms–29 s

Event(s) generated (unless masked away):

When the Read_Connection_Timeout command has completed, a Command Complete event will be generated.

11.2.7.14 Write_Connection_Accept_Timeout

Command	OCF	Command parameters	Return parameters
HCI_Write_Connection_Accept_Timeout	0x0016	Conn_Accept_Timeout	Status

Description:

This command will write the value for the Connection_Accept_Timeout configuration parameter. The Connection_Accept_Timeout configuration parameter allows the Bluetooth hardware to automatically deny a connection request after a specified time interval has occurred and the new connection is not accepted. The parameter defines the time duration from when the Host Controller sends a Connection Request event until the Host Controller will automatically reject an incoming connection.

Command Parameters:

Conn_Accept_Timeout: *Size: 2 Bytes*

Value	Parameter description
N = 0xXXXX	Connection Accept Timeout measured in Number of Baseband slots. Interval Length = *N* * 0.625 ms (1 Baseband slot) Range for *N*: 0x0001–0xB540 Time Range: 0.625 ms–29 s Default: *N* = 0x1F40 Time = 5 s

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 362 of 1169

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Write_Connection_Accept_Timeout command succeeded.
0x01–0xFF	Write_Connection_Accept_Timeout command failed. See Table 109 for list of Error Codes.

Event(s) generated (unless masked away):

When the Write_Connection_Accept_Timeout command has completed, a Command Complete event will be generated.

11.2.7.15 Read_Page_Timeout

Command	OCF	Command parameters	Return parameters
HCI_Read_Page_Timeout	0x0017	—	Status, Page_Timeout

Description:

This command will read the value for the Page_Timeout configuration parameter. The Page_Timeout configuration parameter defines the maximum time the local Link Manager will wait for a baseband page response from the remote device at a locally initiated connection attempt. If this time expires and the remote device has not responded to the page at baseband level, the connection attempt will be considered to have failed.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Read_Page_Timeout command succeeded.
0x01–0xFF	Read_Page_Timeout command failed. See Table 109 for list of Error Codes.

Copyright © 2002 IEEE. All rights reserved.

Page_Timeout: *Size: 2 Bytes*

Value	Parameter description
N = 0xXXXX	Page Timeout measured in Number of Baseband slots. Interval Length = *N* * 0.625 ms (1 Baseband slot) Range for *N*: 0x0001–0xFFFF Time Range: 0.625 ms–40.9 s

Event(s) generated (unless masked away):

When the Read_Page_Timeout command has completed, a Command Complete event will be generated.

11.2.7.16 Write_Page_Timeout

Command	OCF	Command parameters	Return parameters
HCI_Write_Page_Timeout	0x0018	Page_Timeout	Status

Description:

This command will write the value for the Page_Timeout configuration parameter. The Page_Timeout configuration parameter defines the maximum time the local Link Manager will wait for a baseband page response from the remote device at a locally initiated connection attempt. If this time expires and the remote device has not responded to the page at baseband level, the connection attempt will be considered to have failed.

Command Parameters:

Page_Timeout: *Size: 2 Bytes*

Value	Parameter description
0	Illegal Page Timeout. Shall be larger than 0.
N = 0xXXXX	Page Timeout measured in Number of Baseband slots. Interval Length = *N* * 0.625 ms (1 Baseband slot) Range for *N*: 0x0001–0xFFFF Time Range: 0.625 ms–40.9 s Default: *N* = 0x2000 Time = 5.12 s

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Write_Page_Timeout command succeeded.
0x01–0xFF	Write_Page_Timeout command failed. See Table 109 for list of Error Codes.

Copyright © 2002 IEEE. All rights reserved.

LINKPLAY EXHIBIT 1008
Page 364 of 1169

Event(s) generated (unless masked away):

When the Write_Page_Timeout command has completed, a Command Complete event will be generated.

11.2.7.17 Read_Scan_Enable

Command	OCF	Command parameters	Return parameters
HCI_Read_Scan_Enable	0x0019	—	Status, Scan_Enable

Description:

This command will read the value for the Scan_Enable parameter. The Scan_Enable parameter controls whether or not the Bluetooth device will periodically scan for page attempts and/or inquiry requests from other Bluetooth devices. If Page_Scan is enabled, then the device will enter page scan mode based on the value of the Page_Scan_Interval and Page_Scan_Window parameters. If Inquiry_Scan is enabled, then the device will enter Inquiry Scan mode based on the value of the Inquiry_Scan_Interval and Inquiry_Scan_ Window parameters.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Byte*

Value	Parameter description
0x00	Read_Scan_Enable command succeeded.
0x01–0xFF	Read_Scan_Enable command failed. See Table 109 for list of Error Codes.

Scan_Enable: *Size: 1 Byte*

Value	Parameter description
0x00	No Scans enabled.
0x01	Inquiry Scan enabled. Page Scan disabled.
0x02	Inquiry Scan disabled. Page Scan enabled.
0x03	Inquiry Scan enabled. Page Scan enabled.
0x04–0xFF	Reserved

Event(s) generated (unless masked away):

When the Read_Scan_Enable command has completed, a Command Complete event will be generated.

Copyright © 2002 IEEE. All rights reserved.
LINKPLAY EXHIBIT 1008
Page 365 of 1169