STM32WB Bluetooth® Low Energy (BLE) wireless interface

Introduction

Bluetooth® Low Energy (BLE) is a wireless personal area network technology designed and marketed by the Bluetooth® special interest group (Bluetooth® SIG), aimed at novel applications in the healthcare, fitness, beacons, security and home entertainment industries.

Compared to standard Bluetooth®, BLE considerably reduces power consumption and cost while maintaining a similar communication range.

Standard HCI commands are defined in the "Bluetooth specification core V5.2", and the BLE specification is part of it.

All proprietary commands are described in this application note.

1 General information

This document applies to STM32WB Series microcontrollers, based on Arm® cores.

Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

The parameter "size" in the tables of this document is expressed in bytes.

2 ACI/HCI commands

2.1 HCI commands

In Table 1 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / SO, respectively, for Link layer only / Beacon only / Slave only.

Table 1. HCI commands list

Command	OpCode	LO	SO	BO
HCI_DISCONNECT	0x0406	Y	-	-
HCI_READ_REMOTE_VERSION_INFORMATION	0x041D	Y	Y	-
HCI_SET_EVENT_MASK	0x0C01	Y	Y	Y
HCI_RESET	0x0C03	Y	Y	Y
HCI_READ_TRANSMIT_POWER_LEVEL	0x0C2D	Y	Y	-
HCI_SET_CONTROLLER_TO_HOST_FLOW_CONTROL	0x0C31	Y	-	-
HCI_HOST_BUFFER_SIZE	0x0C33	Y	-	-
HCI_HOST_NUMBER_OF_COMPLETED_PACKETS	0x0C35	Y	-	-
HCI_READ_LOCAL_VERSION_INFORMATION	0x1001	Y	Y	Y
HCI_READ_LOCAL_SUPPORTED_COMMANDS	0x1002	Y	Y	Y
HCI_READ_LOCAL_SUPPORTED_FEATURES	0x1003	Y	Y	Y
HCI_READ_BD_ADDR	0x1009	Y	Y	Y
HCI_READ_RSSI	0x1405	Y	Y	Y
HCI_LE_SET_EVENT_MASK	0x2001	Y	Y	Y
HCI_LE_READ_BUFFER_SIZE	0x2002	Y	-	Y
HCI_LE_READ_LOCAL_SUPPORTED_FEATURES	0x2003	Y	Y	Y
HCI_LE_SET_RANDOM_ADDRESS	0x2005	Y	-	Y
HCI_LE_SET_ADVERTISING_PARAMETERS	0x2006	Y	-	Y
HCI_LE_READ_ADVERTISING_CHANNEL_TX_POWER	0x2007	Y	Y	Y
HCI_LE_SET_ADVERTISING_DATA	0x2008	Y	-	Y
HCI_LE_SET_SCAN_RESPONSE_DATA	0x2009	Y	-	Y
HCI_LE_SET_ADVERTISE_ENABLE	0x200A	Y	-	Y
HCI_LE_SET_SCAN_PARAMETERS	0x200B	Y	-	Y
HCI_LE_SET_SCAN_ENABLE	0x200C	Y	-	Y
HCI_LE_CREATE_CONNECTION	0x200D	Y	-	-
HCI_LE_CREATE_CONNECTION_CANCEL	0x200E	Y	-	-
HCI_LE_READ_WHITE_LIST_SIZE	0x200F	Y	-	Y
HCI_LE_CLEAR_WHITE_LIST	0x2010	Y	-	Y
HCI_LE_ADD_DEVICE_TO_WHITE_LIST	0x2011	Y	-	Y
HCI_LE_REMOVE_DEVICE_FROM_WHITE_LIST	0x2012	Y	-	Y
HCI_LE_CONNECTION_UPDATE	0x2013	Y	-	-
HCI_LE_SET_HOST_CHANNEL_CLASSIFICATION	0x2014	Y	-	-
HCI_LE_READ_CHANNEL_MAP	0x2015	Y	Y	-

Command	OpCode	LO	SO	BO
HCI_LE_READ_REMOTE_FEATURES	0x2016	Y	Y	-
HCI_LE_ENCRYPT	0x2017	Y	-	-
HCI_LE_RAND	0x2018	Y	Y	Y
HCI_LE_START_ENCRYPTION	0x2019	Y	-	-
HCI_LE_LONG_TERM_KEY_REQUEST_REPLY	0x201A	Y	-	-
HCI_LE_LONG_TERM_KEY_REQUESTED_NEGATIVE_REPLY	0x201B	Y	-	-
HCI_LE_READ_SUPPORTED_STATES	0x201C	Y	Y	Y
HCI_LE_SET_DATA_LENGTH	0x2022	Y	Y	-
HCI_LE_READ_SUGGESTED_DEFAULT_DATA_LENGTH	0x2023	Y	Y	-
HCI_LE_WRITE_SUGGESTED_DEFAULT_DATA_LENGTH	0x2024	Y	Y	-
HCI_LE_READ_LOCAL_P256_PUBLIC_KEY	0x2025	Y	Y	-
HCI_LE_GENERATE_DHKEY	0x2026	Y	-	-
HCI_LE_ADD_DEVICE_TO_RESOLVING_LIST	0x2027	Y	-	-
HCI_LE_REMOVE_DEVICE_FROM_RESOLVING_LIST	0x2028	Y	-	-
HCI_LE_CLEAR_RESOLVING_LIST	0x2029	Y	-	-
HCI_LE_READ_RESOLVING_LIST_SIZE	0x202A	Y	-	-
HCI_LE_READ_PEER_RESOLVABLE_ADDRESS	0x202B	Y	-	-
HCI_LE_READ_LOCAL_RESOLVABLE_ADDRESS	0x202C	Y	-	-
HCI_LE_SET_ADDRESS_RESOLUTION_ENABLE	0x202D	Y	-	-
HCI_LE_SET_RESOLVABLE_PRIVATE_ADDRESS_TIMEOUT	0x202E	Y	-	-
HCI_LE_READ_MAXIMUM_DATA_LENGTH	0x202F	Y	Y	-
HCI_LE_READ_PHY	0x2030	Y	-	-
HCI_LE_SET_DEFAULT_PHY	0x2031	Y	-	-
HCI_LE_SET_PHY	0x2032	Y	-	-

2.1.1 HCI_DISCONNECT

Description

The HCI_DISCONNECT is used to terminate an existing connection. The Connection_Handle command parameter indicates the connection to be disconnected. The Reason command parameter indicates the reason for ending the connection. The remote controller receives the Reason command parameter in the HCI_DISCONNECTION_COMPLETE_EVENT event. All synchronous connections on a physical link must be disconnected before the ACL connection on the same physical connection is disconnected. It is important to leave a 100 ms blank window before sending any new command (including system hardware reset), since immediately after HCI_DISCONNECTION_COMPLETE_EVENT event, the system could save important information in non-volatile memory

Input parameters

Table 2. HCI_DISCONNECT input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Reason	1	The reason for ending the connection.	• 0x05: Authentication failure • 0x13: Remote user terminated connection • 0x14: Remote device terminated connection due to low resources • 0x15: Remote device terminated connection due to power off • 0x1A: Unsupported remote feature • 0x3B: Unacceptable connection parameters

Output parameters

Table 3. HCI_DISCONNECT output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_DISCONNECTION_COMPLETE_EVENT

2.1.2 HCI_READ_REMOTE_VERSION_INFORMATION

Description

This command obtains the values for the version information for the remote device identified by the Connection_Handle parameter. The Connection_Handle must be a Connection_Handle for an ACL or LE connection.

Input parameters

Table 4. HCI_READ_REMOTE_VERSION_INFORMATION input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the Connection_Handle version information to get.	0x0000 ... 0x0EFF

Output parameters

Table 5. **HCI_READ_REMOTE_VERSION_INFORMATION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_READ_REMOTE_VERSION_INFORMATION_COMPLETE_EVENT

2.1.3 HCI_SET_EVENT_MASK

Description

The HCI_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 1, then the event associated with that bit is enabled. For an LE controller, the LE Meta event bit in the Event_Mask enables or disables all LE events in the LE Meta event. The host has to deal with each occurring event. The event mask allows the host to control how much it is interrupted.

Input parameters

Table 6. HCI_SET_EVENT_MASK input parameters

Parameter	Size	Description	Possible values
Event_Mask	8	Event mask. Default: 0x20001FFFFFFFFFFF	Bitmask of: • 0x0000000000000000: No events specified • 0x0000000000000010: Disconnection complete event • 0x0000000000000080: Encryption change event • 0x0000000000000800: Read remote version information complete event • 0x0000000000008000: Hardware error event • 0x0000080000000000: Encryption key refresh complete event • 0x2000000000000000: LE Meta-event

Output parameters

Table 7. HCI_SET_EVENT_MASK output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.4 HCI_RESET

Description

This command resets the link layer on an LE controller. It does not affect the used HCI transport layer since the HCI transport layers may have reset mechanisms of their own. After the reset is completed, the current operational state is lost, the controller enters standby mode and automatically reverts to the default values for the parameters for which default values are defined in the specification.

Note: The Reset command not necessarily performs a hardware reset. This is defined implementation. The host does not send additional HCI commands before the command complete event related to the reset command has been received.

Input parameters

None

Output parameters

Table 8. HCI_RESET output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.5 HCI_READ_TRANSMIT_POWER_LEVEL

Description

This command reads the values for the Transmit_Power_Level parameter for the specified Connection_Handle. The Connection_Handle is a Connection_Handle for an ACL connection.

Input parameters

Table 9. HCI_READ_TRANSMIT_POWER_LEVEL input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies which Connection_Handle's transmit power level setting to read.	0x0000 ... 0x0EFF
Type	1	Current or maximum transmit power level.	• 0x00: Read current transmit power level • 0x01: Read maximum transmit power level

Output parameters

Table 10. HCI_READ_TRANSMIT_POWER_LEVEL output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Connection_Handle	2	Connection handle related to the response	0x0000 ... 0x0EFF
Transmit_Power_Level	1	Size: 1 Octet (signed integer) Units: dBm	-30 … 20

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.6 HCI_SET_CONTROLLER_TO_HOST_FLOW_CONTROL

Description

This command is used by the Host to turn flow control on or off for data and/or voice sent from the controller to the host. If flow control is turned off, the host does not send the Host_Number_Of_Completed_Packets command. That command is ignored by the controller if it is sent by the host and flow control is off.

If flow control is turned on for HCI ACL data packets and off for HCI synchronous data packets, Host_Number_Of_Completed_Packets commands sent by the host must only contain Connection_Handles for ACL connections. If flow control is turned off for HCI ACL data packets and on for HCI synchronous data packets, Host_Number_Of_Completed_Packets commands sent by the host must only contain Connection_Handles for synchronous connections.

If flow control is turned on for HCI ACL Data Packets and HCI synchronous data packets, the host sends Host_Number_Of_Completed_Packets commands both for ACL connections and synchronous connections. The Flow_Control_Enable parameter only is changed if no connections exist.

Input parameters

Table 11. HCI_SET_CONTROLLER_TO_HOST_FLOW_CONTROL input parameters

Parameter	Size	Description	Possible values
Flow_Control_Enable	1	Enable/Disable the flow control	• 0x00: Flow control off in direction from controller to host. Default • 0x01: Flow control on for HCI ACL data packets and off for HCI synchronous. Data packets in direction from controller to host. • 0x02: Flow control off for HCI ACL data packets and on for HCI synchronous. Data packets in direction from controller to host. • 0x03: Flow control on both for HCI ACL data packets and HCI synchronous. Data packets in direction from controller to host

Output parameters

Table 12. HCI_SET_CONTROLLER_TO_HOST_FLOW_CONTROL output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT

2.1.7 HCI_HOST_BUFFER_SIZE

Description

The Host_Buffer_Size command is used by the host to notify the controller about the maximum size of the data portion of HCI ACL and synchronous data packets sent from the controller to the host. The controller segments the data to be transmitted to the host according to their size, so that the HCI data packets contain data with up to these sizes.

The Host_Buffer_Size command also notifies the controller about the total number of HCI ACL and synchronous data packets stored in the data buffers of the host. If flow control from the controller to the host is turned off, and the Host_Buffer_Size command has not been issued by the host, this means that the controller sends HCI data packets to the host with any lengths the controller wants to use, and it is assumed that the data buffer sizes of the host are unlimited. If flow control from the controller to the host is turned on, the Host_Buffer_Size command, after a power-on or a reset, always is sent by the host before the first Host_Number_Of_Completed_Packets command is sent.

The set controller to host flow control command is used to turn flow control on or off.

The Host_ACL_Data_Packet_Length command parameter is used to determine the size of the L2CAP segments contained in ACL data packets, which are transferred from the controller to the host.

The Host_Synchronous_Data_Packet_Length command parameter is used to determine the maximum size of HCI synchronous data packets. Both the host and the controller support command and event packets, where the data portion (excluding header) contained in the packets is 255 octets in size.

The Host_Total_Num_ACL_Data_Packets command parameter contains the total number of HCI ACL data packets stored in the data buffers of the host. The controller determines how the buffers are to be divided between different Connection_Handles.

The Host_Total_Num_Synchronous_ Data_Packets command parameter gives the same information for HCI synchronous Data Packets.

Note: The Host_ACL_Data_Packet_Length and Host_Synchronous_Data_ Packet_Length command parameters do not include the length of the HCI Data Packet header.

Input parameters

Table 13. HCI_HOST_BUFFER_SIZE input parameters

Parameter	Size	Description	Possible values
Host_ACL_Data_Packet_Length	2	Maximum length (in octets) of the data portion of each HCI ACL data packet that the Host is able to accept. Must be greater or equal to 251 bytes	-
Host_Synchronous_Data_Packet_Length	1	Maximum length (in octets) of the data portion of each HCI synchronous data packet that the host is able to accept.	-
Host_Total_Num_ACL_Data_Packets	2	Total number of HCI ACL data packets that can be stored in the data buffers of the host.	-
Host_Total_Num_Synchronous_Data_Packets	2	Total number of HCI synchronous data packets that can be stored in the data buffers of the host.	-

Output parameters

Table 14. HCI_HOST_BUFFER_SIZE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT

2.1.8 HCI_HOST_NUMBER_OF_COMPLETED_PACKETS

Description

The Host_Number_Of_Completed_Packets command is used by the host to indicate to the controller the number of HCI data packets that have been completed for each Connection_Handle since the previous Host_Number_Of_ Completed_Packets command was sent to the controller. This means that the corresponding buffer space has been freed in the host.

Based on this information, and the Host_Total_Num_ACL_Data_Packets and Host_Total_Num_Synchronous_Data_Packets command parameters of the Host_Buffer_Size command, the controller determines for which Connection_Handles the following HCI data packets must be sent to the host. The command is issued only by the host if flow control in the direction from the controller to the host is on and there is at least one connection, or if the controller is in local loop-back mode. Otherwise, the command is ignored by the controller. When the host has completed one or more HCI data packet(s) it sends a Host_Number_Of_Completed_Packets command to the controller, until it finally reports that all pending HCI data packets have been completed. The frequency at which this command is sent is manufacturer specific.

The set controller to host flow control command is used to turn flow control on or off. If flow control from the controller to the host is turned on, the Host_Buffer_Size command always is sent by the host after a power-on or a reset before the first Host_Number_Of_Completed_Packets command is sent.

Note: The Host_Number_Of_Completed_Packets command is a special command in the sense that no event is normally generated after the command has completed. The command may be sent at any time by the host when there is at least one connection, or if the controller is in local loopback mode independent of other commands. The normal flow control for commands is not used for the Host_Number_Of_Completed_Packets command.

Input parameters

Table 15. **HCI_HOST_NUMBER_OF_COMPLETED_PACKETS input parameters**

Parameter	Size	Description	Possible values
Number_Of_Handles	1	The number of Connection_Handles and Host_Num_Of_Completed_Packets parameters pairs contained in this command.	0 - 255
Connection_Handle[i]	Number_Of_Handles * 2	Connection_Handle	0x0000-0x0EFF
Host_Num_Of_Completed_Packets[i]	Number_Of_Handles * 2	The number of HCI data packets that have been completed for the associated Connection_Handle since the previous time the event was returned.	0x0000-0xFFFF

Output parameters

None

Events generated

Event(s) generated (unless masked away): normally, no event is generated after the Host_Number_Of_Completed_Packets command has completed. However, if the Host_Number_Of_Completed_Packets command contains one or more invalid parameters, the controller returns a command complete event with a failure status indicating the invalid HCI command parameters error code. The host may send the Host_Number_Of_Completed_Packets command at any time when there is at least one connection, or if the controller is in local loopback mode. The normal flow control for commands is not used for this command.

2.1.9 HCI_READ_LOCAL_VERSION_INFORMATION

Description

This command reads the values for the version information for the local controller. The HCI version information defines the version information of the HCI layer. The LMP/PAL version information defines the version of the LMP or PAL. The Manufacturer_Name information indicates the manufacturer of the local device. The HCI revision and LMP/PAL subversion are implementation dependent.

Input parameters

None

Output parameters

Table 16. **HCI_READ_LOCAL_VERSION_INFORMATION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
HCI_Version	1	See Bluetooth assigned numbers	-
HCI_Revision	2	Revision of the current HCI in the BR/EDR controller.	-
LMP_PAL_Version	1	Version of the current LMP or PAL in the controller.	-
Manufacturer_Name	2	Manufacturer name of the BR/EDR controller.	-
LMP_PAL_Subversion	2	Subversion of the current LMP or PAL in the controller. This value is implementation dependent.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.10 HCI_READ_LOCAL_SUPPORTED_COMMANDS

Description

This command reads the list of HCI commands supported for the local controller. This command returns the Supported_Commands configuration parameter. It is implied that if a command is listed as supported, the feature underlying that command is also supported.

Input parameters

None

Output parameters

Table 17. HCI_READ_LOCAL_SUPPORTED_COMMANDS output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Supported_Commands	64	Bit mask for each HCI command. If a bit is 1, the controller supports the corresponding command and the features required for the command. Unsupported or undefined commands is set to 0.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.11 HCI_READ_LOCAL_SUPPORTED_FEATURES

Description

This command requests a list of the supported features for the local controller. This command returns a list of the LMP features.

Input parameters

None

Output parameters

Table 18. HCI_READ_LOCAL_SUPPORTED_FEATURES output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
LMP_Features	8	Bit mask list of LMP features.	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.12 HCI_READ_BD_ADDR

Description

On an LE Controller, this command reads the public device address. If this controller does not have a public device address, the value 0x000000000000 is returned. On an LE controller, the public address is the same as the BD_ADDR.

Input parameters

None

Output parameters

Table 19. HCI_READ_BD_ADDR output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
BD_ADDR	6	BD_ADDR (Bluetooth device address) of the device	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.13 HCI_READ_RSSI

Description

This command reads the received signal strength indication (RSSI) value from a controller. For an LE transport, a Connection_Handle is used as the handle command parameter and return parameter. The meaning of the RSSI metric is an absolute receiver signal strength value in dBm to ±6 dB accuracy. If the RSSI cannot be read, the RSSI metric is set to 127.

Input parameters

Table 20. HCI_READ_RSSI input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 21. HCI_READ_RSSI output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
RSSI	1	N Size: 1 Octet (signed integer) Units: dBm	• 127: RSSI not available • -127 ... 20

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.14 HCI_LE_SET_EVENT_MASK

Description

The LE_Set_Event_Mask command is used to control which LE events are generated by the HCI for the host. If the bit in the LE_Event_Mask is set to 1, then the event associated with that bit is enabled. The host has to deal with each event generated by an LE controller. The event mask allows the host to control which events interrupt it. For LE events to be generated, the LE Meta-Event bit in the Event_Mask also is set. If that bit is not set, then LE events are not generated, regardless of how the LE_Event_Mask is set.

Input parameters

Table 22. HCI_LE_SET_EVENT_MASK input parameters

Parameter	Size	Description	Possible values
LE_Event_Mask	8	LE event mask. Default: 0x00000000000FFFFF	Bitmask of: • 0x0000000000000000: No LE events specified • 0x0000000000000001: LE connection complete event • 0x0000000000000002: LE advertising report event • 0x0000000000000004: LE connection update complete event • 0x0000000000000008: LE read remote used features complete event • 0x0000000000000010: LE long term key request event • 0x0000000000000020: LE remote connection parameter request event • 0x0000000000000040: ILE data length change event • 0x0000000000000080: LE read local P-256 public key complete event • 0x0000000000000100: LE generate DHKey complete event • 0x0000000000000200: LE enhanced connection complete event • 0x0000000000000400: LE direct advertising report event • 0x0000000000000800: LE PHY update complete event • 0x0000000000001000: LE extended advertising report event • 0x0000000000002000: LE periodic advertising sync established event • 0x0000000000004000: LE periodic advertising report event • 0x0000000000008000: LE periodic advertising sync lost event • 0x0000000000010000: LE extended scan timeouout event • 0x0000000000020000: LE extended advertising set terminated event • 0x0000000000040000: LE scan request received event • 0x0000000000080000: LE channel selection algorithm event

Output parameters

Table 23. HCI_LE_SET_EVENT_MASK output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.15 HCI_LE_READ_BUFFER_SIZE

Description

The LE_Read_Buffer_Size command is used to read the maximum size of the data portion of HCI LE ACL data packets sent from the host to the controller. The host segments the data transmitted to the controller according to these values, so that the HCI data packets contain data with up to this size. The LE_Read_Buffer_Size command also returns the total number of HCI LE ACL data packets that is stored in the data buffers of the controller. The LE_Read_Buffer_Size command must be issued by the host before it sends any data to an LE controller. If the controller returns a length value of zero, the host uses the Read_Buffer_Size command to determine the size of the data buffers.

Note: Both the Read_Buffer_Size and LE_Read_Buffer_Size commands may return buffer length and number of packets parameter values that are non zero.

The HC_LE_ACL_Data_Packet_Length return parameter is used to determine the size of the L2CAP PDU segments contained in ACL data packets, which are transferred from the host to the controller to be broken up into packets by the link layer. Both the host and the controller support command and event packets, where the data portion (excluding header) contained in the packets is 255 octets in size. The HC_Total_Num_LE_ACL_Data_Packets return parameter contains the total number of HCI ACL data packets that is stored in the data buffers of the controller. The host determines how the buffers are to be divided between different connection handles.

Note: The HC_LE_ACL_Data_Packet_Length return parameter does not include the length of the HCI data packet header.

Input parameters

None

Output parameters

Table 24. HCI_LE_READ_BUFFER_SIZE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
HC_LE_ACL_Data_Packet_Length	2	• 0x0000 No dedicated LE buffer, use Read_Buffer_Size command. 0x0001 • 0xFFFF Maximum length (in octets) of the data portion of each HCI ACL data packet that the controller is able to accept	-
HC_Total_Num_LE_ACL_Data_Packets	1	• 0x00 No dedicated LE buffer, use Read_Buffer_Size command • 0x01 - 0xFF total number of HCI ACL data packets that is stored in the data buffers of the controller	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.16 HCI_LE_READ_LOCAL_SUPPORTED_FEATURES

Description

This command requests the list of the supported LE features for the controller.

Input parameters

None

Output parameters

Table 25. **HCI_LE_READ_LOCAL_SUPPORTED_FEATURES output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
LE_Features	8	Bit mask list of LE features	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.17 HCI_LE_SET_RANDOM_ADDRESS

Description

The LE_Set_Random_Address command is used by the host to set the LE random device address in the controller.

Input parameters

Table 26. **HCI_LE_SET_RANDOM_ADDRESS input parameters**

Parameter	Size	Description	Possible values
Random_Address	6	Random device address	-

Output parameters

Table 27. **HCI_LE_SET_RANDOM_ADDRESS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.18 HCI_LE_SET_ADVERTISING_PARAMETERS

Description

The LE_Set_Advertising_Parameters command is used by the host to set the advertising parameters. The Advertising_Interval_Min is lower than or equal to the Advertising_Interval_Max. The Advertising_Interval_Min and Advertising_Interval_Max must not be the same value to enable the controller to determine the best advertising interval given other activities.

For high duty cycle directed advertising, i.e. when Advertising_Type is 0x01 (ADV_DIRECT_IND, high duty cycle), the Advertising_Interval_Min and Advertising_Interval_Max parameters are not used and are ignored. The Advertising_Type is used to determine the packet type used for advertising when it is enabled. The Advertising_Interval_Min and Advertising_Interval_Max is not set to less than 0x00A0 (100 ms) if the Advertising_Type is set to 0x02 (ADV_SCAN_IND) or 0x03 (ADV_NONCONN_IND).

The Own_Address_Type determines if the advertising packets are identified with the public device address of the device, or a random device address as written by the LE_Set_Random_Address command. If directed advertising is performed, i.e. when Advertising_Type is set to 0x01.

(ADV_DIRECT_IND, high duty cycle) or 0x04 (ADV_DIRECT_IND, low duty cycle mode), then the Direct_Address_Type and Direct_Address are valid, otherwise they are ignored by the controller and not used. The Advertising_Channel_Map is a bit field that indicates the advertising channels used when transmitting advertising packets. At least one channel bit is set in the Advertising_Channel_Map parameter. The Advertising_Filter_Policy parameter is ignored when directed advertising is enabled. The host not issues this command when advertising is enabled in the controller; if it is, the command disallowed error code is used.

Input parameters

Table 28. HCI_LE_SET_ADVERTISING_PARAMETERS input parameters

Parameter	Size	Description	Possible values
Advertising_Interval_Min	2	Minimum advertising interval. Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Advertising_Type	1	Advertising type	• 0x00: ADV_IND (connectable undirected advertising) • 0x01: ADV_DIRECT_IND, high duty cycle (connectable high duty cycle directed advertising) • 0x02: ADV_SCAN_IND (scannable undirected advertising) • 0x03: ADV_NONCONN_IND (non-connectable undirected advertising) • 0x04: ADV_DIRECT_IND, low duty cycle (connectable low duty cycle directed advertising)
Own_Address_Type	1	Own address type. • 0x00: Public device address • 0x01: Random device address • 0x02: Controller generates resolvable private address based on the local IRK from resolving list. If resolving list contains no matching entry, use public address. • 0x03: Controller generates resolvable private address based on the local IRK from resolving list. If resolving list contains no matching entry, use random address from LE_Set_Random_Address.	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address or public address • 0x03: Resolvable private address or random address
Peer_Address_Type	1	The address type of the peer device	• 0x00: Public device address • 0x01: Random device address
Peer_Address	6	Public device address, random device address, public identity address or random (static) identity address of the device to be connected.	-
Advertising_Channel_Map	1	Advertising channel map. Default: 00000111b (all channels enabled).	Bitmask of: • 0x01: ch 37 • 0x02: ch 38 • 0x04: ch 39
Advertising_Filter_Policy	1	Advertising filter policy.	• 0x00: Allow scan request from any, allow connect request from any • 0x01: Allow scan request from white list only, allow connect request from any • 0x02: Allow scan request from any, allow connect request from white list only • 0x03: Allow scan request from white list only, allow connect request from white list only

Output parameters

Table 29. **HCI_LE_SET_ADVERTISING_PARAMETERS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.19 HCI_LE_READ_ADVERTISING_CHANNEL_TX_POWER

Description

The LE_Read_Advertising_Channel_Tx_Power command is used by the host to read the transmit power level used for LE advertising channel packets.

Input parameters

None

Output parameters

Table 30. **HCI_LE_READ_ADVERTISING_CHANNEL_TX_POWERutput parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Transmit_Power_Level	1	Size: 1 octet (signed integer) Units: dBm Accuracy: ± 4 dBm	-20 ...10

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.20 HCI_LE_SET_ADVERTISING_DATA

Description

The LE_Set_Advertising_Data command is used to set the data used in advertising packets that have a data field. Only the significant part of the Advertising_Data is transmitted in the advertising packets.

Input parameters

Table 31. **HCI_LE_SET_ADVERTISING_DATA input parameters**

Parameter	Size	Description	Possible values
Advertising_Data_Length	1	The number of significant octets in the following data field.	-
Advertising_Data	31	31 octets of data formatted.	-

Output parameters

Table 32. **HCI_LE_SET_ADVERTISING_DATA output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.21 HCI_LE_SET_SCAN_RESPONSE_DATA

Description

This command is used to provide data used in scanning packets that have a data field. Only the significant part of the Scan_Response_Data is transmitted in the scanning packets.

Input parameters

Table 33. HCI_LE_SET_SCAN_RESPONSE_DATA input parameters

Parameter	Size	Description	Possible values
Scan_Response_Data_Length	1	The number of significant octets in the following data field	-
Scan_Response_Data	31	31 octets of data formatted	-

Output parameters

Table 34. HCI_LE_SET_SCAN_RESPONSE_DATA output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.22 HCI_LE_SET_ADVERTISE_ENABLE

Description

The LE_Set_Advertise_Enable command is used to request the controller to start or stop advertising. The controller manages the timing of advertisements according to theparameters given in the LE_Set_Advertising_Parameters command. The controller continues advertising until the host issues an LE_Set_Advertise_Enable command with Advertising_Enable set to 0x00 (advertising is disabled) or until a connection is created or until the advertising is timed out due to high duty cycle directed advertising. In these cases, advertising is then disabled.

Input parameters

Table 35. HCI_LE_SET_ADVERTISE_ENABLE input parameters

Parameter	Size	Description	Possible values
Advertising_Enable	1	Enable/disable advertise. Default is 0 (disabled).	• 0x00: Advertising is disabled • 0x01: Advertising is enabled

Output parameters

Table 36. HCI_LE_SET_ADVERTISE_ENABLE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- HCI_LE_CONNECTION_COMPLETE_EVENT

2.1.23 HCI_LE_SET_SCAN_PARAMETERS

Description

The LE_Set_Scan_Parameters command is used to set the scan parameters. The LE_Scan_Type parameter controls the type of scan to perform. The LE_Scan_Interval and LE_Scan_Window parameters are recommendations from the host on how long (LE_Scan_Window) and how frequently (LE_Scan_Interval) the controller must scan.

The LE_Scan_Window parameter always is set to a value smaller or equal to the value set for the LE_Scan_Interval parameter. If they are set to the same value scanning must be run continuously. The Own_Address_Type parameter determines the address used (public or random device address) when performing active scan. The host does not issue this command when scanning is enabled in the controller; if it is, the command disallowed error code is used.

Input parameters

Table 37. HCI_LE_SET_SCAN_PARAMETERS input parameters

Parameter	Size	Description	Possible values
LE_Scan_Type	1	Passive or active scanning. With active scanning SCAN_REQ packets are sent.	• 0x00: Passive scanning • 0x01: Active scanning
LE_Scan_Interval	2	This is defined as the time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	• 0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window is less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	• 0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
Own_Address_Type	1	Own address type. • 0x00: Public device address • 0x01: Random device address • 0x02: Controller generates resolvable private address based on the local IRK from resolving list. If resolving list contains no matching entry, use public address. • 0x03: Controller generates resolvable private address based on the local IRK from resolving list. If resolving list contains no matching entry, use random address from LE_Set_Random_Address	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address or public address • 0x03: Resolvable private address or random address
Scanning_Filter_Policy	1	• 0x00 Accept all advertisement packets. Directed advertising packets which are not addressed for this device is ignored. • 0x01 ignore advertisement packets from devices not in the white list only. Directed advertising packets which are not addressed for this device is ignored • 0x02 accept all undirected advertisement packets. Directed advertisement packets where initiator address is a RPA and directed advertisement packets addressed to this device is accepted. • 0x03 Accept all undirected advertisement packets from devices that are in the white List. Directed advertisement packets where initiator address is RPA and directed advertisement packets addressed to this device is accepted.	• 0x00: Accept all • 0x01: Ignore devices not in the white list • 0x02: Accept all (use resolving list) • 0x03: Ignore devices not in the white list (use resolving list)

Output parameters

Table 38. **HCI_LE_SET_SCAN_PARAMETERS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.24 HCI_LE_SET_SCAN_ENABLE

Description

The LE_Set_Scan_Enable command is used to start scanning. Scanning is used to discover advertising devices nearby. The Filter_Duplicates parameter controls whether the link layer filters duplicate advertising reports to the host, or if the link layer must generate advertising reports for each packet received.

Input parameters

Table 39. **HCI_LE_SET_SCAN_ENABLE input parameters**

Parameter	Size	Description	Possible values
LE_Scan_Enable	1	Enable/disable scan. Default is 0 (disabled)	• 0x00: Scanning disabled • 0x01: Scanning enabled
Filter_Duplicates	1	Enable/disable duplicate filtering	• 0x00: Duplicate filtering disabled • 0x01: Duplicate filtering enabled

Output parameters

Table 40. **HCI_LE_SET_SCAN_ENABLE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- HCI_LE_ADVERTISING_REPORT_EVENT

2.1.25 HCI_LE_CREATE_CONNECTION

Description

The LE_Create_Connection command is used to create a link layer connection to a connectable advertiser. The LE_Scan_Interval and LE_Scan_Window parameters are recommendations from the host on how long (LE_Scan_Window) and how frequently (LE_Scan_Interval) the controller must scan. The LE_Scan_Window parameter is set to a value smaller than or equal to the value set for the LE_Scan_Interval parameter. If both are set to the same value, scanning runs continuously. The Initiator_Filter_Policy is used to determine whether the white list is used. If the white list is not used, the Peer_Address_Type and the Peer_Address parameters specify the address type and address of the advertising device to connect to. The link layer sets the address in the CONNECT_REQ packets to either the public device address or the random device addressed based on the Own_Address_Type parameter. The Conn_Interval_Min and Conn_Interval_Max parameters define the minimum and maximum allowed connection interval. The Conn_Interval_Min parameter is not greater than the Conn_Interval_Max parameter. The Conn_Latency parameter defines the maximum allowed connection latency. The Supervision_Timeout parameter defines the link supervision timeout for the connection. The Supervision_Timeout in milliseconds is larger than (1 + Conn_Latency) * Conn_Interval_Max * 2, where Conn_Interval_Max is given in milliseconds. The Minimum_CE_Length and Maximum_CE_Length parameters are informative parameters providing the controller with the expected minimum and maximum length of the connection events. The Minimum_CE_Length parameter is less than or equal to the Maximum_CE_Length parameter. The host does not issue this command when another LE_Create_Connection is pending in the controller; if this occurs the controller returns the command disallowed error code is used.

Input parameters

Table 41. HCI_LE_CREATE_CONNECTION input parameters

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	This is defined as the time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window is less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
Initiator_Filter_Policy	1	• 0x00 white list is not used to determine which advertiser to connect to. Peer_Address_Type and Peer_Address is used • 0x01 white list is used to determine which advertiser to connect to. Peer_Address_Type and Peer_Address is ignored	• 0x00: White list not used • 0x01: White list used
Peer_Address_Type	1	• 0x00 Public device address • 0x01 Random device address • 0x02 Public identity address (corresponds to resolved private address) • 0x03 Random (static) identity address (corresponds to resolved private address)	• 0x00: Public device address • 0x01: Random device address • 0x02: Public identity address • 0x03: Random (static) identity address
Peer_Address	1	Public device address or random device address of the device to be connected.	-
Own_Address_Type	1	Own address type. • 0x00: Public device address • 0x01 Random device address • 0x02: Controller generates resolvable private address based on the local IRK from resolving list. If resolving list contains no matching entry, use public address. • 0x03: Controller generates resolvable private address based on the local IRK from resolving list. If resolving list contains no matching entry, use random address from LE_Set_Random_Address.	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address or public address • 0x03: Resolvable private address or random address
Conn_Interval_Min	2	Minimum value for the connection event interval. This is less than or equal to Conn_Interval_Max. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Interval_Max	2	Maximum value for the connection event interval. This is greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Slave latency for the connection in number of connection events	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE Link. It is a multiple of 10 ms and larger than (1 + connSlaveLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)
Minimum_CE_Length	2	Information parameter about the minimum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Maximum_CE_Length	2	Information parameter about the maximum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)

Output parameters

Table 42. **HCI_LE_CREATE_CONNECTION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_COMMAND_COMPLETE_EVENT

2.1.26 HCI_LE_CREATE_CONNECTION_CANCEL

Description

The LE_Create_Connection_Cancel command is used to cancel the LE_Create_Connection command. This command is only issued after the LE_Create_Connection command has been issued, a command status event has been received for the LE create connection command and before the LE connection complete event.

Input parameters

None

Output parameters

Table 43. **HCI_LE_CREATE_CONNECTION_CANCEL output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- HCI_LE_CONNECTION_COMPLETE_EVENT

2.1.27 HCI_LE_READ_WHITE_LIST_SIZE

Description

The LE_Read_White_List_Size command is used to read the total number of white list entries that can be stored in the controller.

Input parameters

None

Output parameters

Table 44. **HCI_LE_READ_WHITE_LIST_SIZE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
White_List_Size	1	Total number of white list entries that can be stored in the controller.	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.28 HCI_LE_CLEAR_WHITE_LIST

Description

The LE_Clear_White_List command is used to clear the white list stored in the controller. This command can be used at any time except when: - the advertising filter policy uses the white list and advertising is enabled. - the scanning filter policy uses the white list and scanning is enabled. - the initiator filter policy uses the white list and an LE_Create_Connection command is outstanding.

Input parameters

None

Output parameters

Table 45. HCI_LE_CLEAR_WHITE_LIST output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.29 HCI_LE_ADD_DEVICE_TO_WHITE_LIST

Description

The LE_Add_Device_To_White_List command is used to add a single device to the white list stored in the controller. This command can be used at any time except when:

- The advertising filter policy uses the white list and advertising is enabled
- The scanning filter policy uses the white list and scanning is enabled
- The initiator filter policy uses the white list and a create connection command is outstanding

Input parameters

Table 46. HCI_LE_ADD_DEVICE_TO_WHITE_LIST input parameters

Parameter	Size	Description	Possible values
Address_Type	1	Address type	• 0x00: Public device address • 0x01: Random device address
Address	6	Public device address or random device address of the device to be added to the white list.	-

Output parameters

Table 47. HCI_LE_ADD_DEVICE_TO_WHITE_LIST output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.30 HCI_LE_REMOVE_DEVICE_FROM_WHITE_LIST

Description

The LE_Remove_Device_From_White_List command is used to remove a single device from the white list stored in the controller. This command can be used at any time except when:

- The advertising filter policy uses the white list and advertising is enabled
- The scanning filter policy uses the white list and scanning is enabled
- The initiator filter policy uses the white list and a create connection command is outstanding

Input parameters

Table 48. **HCI_LE_REMOVE_DEVICE_FROM_WHITE_LIST input parameters**

Parameter	Size	Description	Possible values
Address_Type	1	Address type	• 0x00: Public device address • 0x01: Random device address
Address	6	Public device address or random device address of the device to be removed from the white list	-

Output parameters

Table 49. **HCI_LE_REMOVE_DEVICE_FROM_WHITE_LIST output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.31 HCI_LE_CONNECTION_UPDATE

Description

The LE_Connection_Update command is used to change the link layer connection parameters of a connection. This command is supported only on master side. The Conn_Interval_Min and Conn_Interval_Max parameters are used to define the minimum and maximum allowed connection interval. The Conn_Interval_Min parameter is not greater than the Conn_Interval_Max parameter. The Conn_Latency parameter defines the maximum allowed connection latency. The Supervision_Timeout parameter defines the link supervision timeout for the LE link. The Supervision_Timeout in milliseconds is larger than (1 + Conn_Latency) * Conn_Interval_Max * 2, where Conn_Interval_Max is given in milliseconds. The Minimum_CE_Length and Maximum_CE_Length are information parameters providing the controller with a hint about the expected minimum and maximum length of the connection events. The Minimum_CE_Length is less than or equal to the Maximum_CE_Length. The actual parameter values selected by the link layer may be different from the parameter values provided by the host through this command.

Input parameters

Table 50. **HCI_LE_CONNECTION_UPDATE input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Conn_Interval_Min	2	Minimum value for the connection event interval. Thisis less than or equal to Conn_Interval_Max. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Interval_Max	2	Maximum value for the connection event interval. This is greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE Link. It is a multiple of 10 ms and larger than (1 + connSlaveLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)
Minimum_CE_Length	2	Information parameter about the minimum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Maximum_CE_Length	2	Information parameter about the maximum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)

Output parameters

Table 51. **HCI_LE_CONNECTION_UPDATE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_CONNECTION_COMPLETE_EVENT

2.1.32 HCI_LE_SET_HOST_CHANNEL_CLASSIFICATION

Description

The LE_Set_Host_Channel_Classification command allows the host to specify a channel classification for data channels based on its "local information".

This classification persists until overwritten with a subsequent LE_Set_Host_Channel_Classification command or until the controller is reset using the reset comman. If this command is used, the host must send it within 10 seconds of knowing that the channel classification has changed. The interval between two successive commands sent is at least one second. This command is only used when the local device supports the master role.

Input parameters

Table 52. **HCI_LE_SET_HOST_CHANNEL_CLASSIFICATION input parameters**

Parameter	Size	Description	Possible values
LE_Channel_Map	5	This parameter contains 37 1-bit fields. The n^{th} such field (in the range 0 to 36) contains the value for the link layer channel index n. Channel n is bad = 0. Channel n is unknown = 1. The most significant bits are reserved and is set to 0. At least one channel is marked as unknown.	-

Output parameters

Table 53. **HCI_LE_SET_HOST_CHANNEL_CLASSIFICATION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.33 HCI_LE_READ_CHANNEL_MAP

Description

The LE_Read_Channel_Map command returns the current Channel_Map for the specified Connection_Handle. The returned value indicates the state of the Channel_Map specified by the last transmitted or received Channel_Map (in a CONNECT_REQ or LL_CHANNEL_MAP_REQ message) for the specified Connection_Handle, regardless of whether the Master has received an acknowledgement.

Input parameters

Table 54. **HCI_LE_READ_CHANNEL_MAP input parameters**

Parameter	Size		Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 55. **HCI_LE_READ_CHANNEL_MAP output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Connection_Handle	2	Connection handle related to the response	0x0000 ... 0x0EFF
LE_Channel_Map	5	This parameter contains 37 1-bit fields. The n^{th} such field (in the range 0 to 36) contains the value for the link layer channel index n. Channel n is unused = 0. Channel n is used = 1. The most significant bits are reserved and is set to 0.	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.34 HCI_LE_READ_REMOTE_FEATURES

Description

This command requests a list of the used LE features from the remote device. This command returns a list of the used LE features. This command may be issued on both the master and slave.

Input parameters

Table 56. HCI_LE_READ_REMOTE_FEATURES input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 57. HCI_LE_READ_REMOTE_FEATURES output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- HCI_LE_READ_REMOTE_FEATURES_COMPLETE_EVENT

2.1.35 HCI_LE_ENCRYPT

Description

The LE_Encrypt command is used to request the controller to encrypt the Plaintext_Data in the command using the key given in the command and returns the Encrypted_Data to the host.

Input parameters

Table 58. HCI_LE_ENCRYPT input parameters

Parameter	Size	Description	Possible values
Key	16	128 bit key for the encryption of the data given in the command.	-
Plaintext_Data	16	128 bit data block that is requested to be encrypted.	-

Output parameters

Table 59. HCI_LE_ENCRYPT output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Encrypted_Data	16	128 bit encrypted data block.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.36 HCI_LE_RAND

Description

The LE_Rand command is used to request the controller to generate eight octets of random data to be sent to the host.

Input parameters

None

Output parameters

Table 60. **HCI_LE_RAND output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Random_Number	8	Random number	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.37 HCI_LE_START_ENCRYPTION

Description

The LE_Start_Encryption command is used to authenticate the given encryption key associated with the remote device specified by the connection handle, and once authenticated encrypts the connection. If the connection is already encrypted then the controller pauses connection encryption before attempting to authenticate the given encryption key, and then re-encrypts the connection. While encryption is paused no user data is transmitted. On an authentication failure, the connection is automatically disconnected by the link layer. If this command succeeds, then the connection is encrypted. This command is only used when the local device's role is master.

Input parameters

Table 61. **HCI_LE_START_ENCRYPTION input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Random_Number	8	64 bit random number	-
Encrypted_Diversifier	2	16 bit encrypted diversifier	-
Long_Term_Key	16	128 bit long term key	-

Output parameters

Table 62. **HCI_LE_START_ENCRYPTION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_ENCRYPTION_CHANGE_EVENT

2.1.38 HCI_LE_LONG_TERM_KEY_REQUEST_REPLY

Description

The LE_Long_Term_Key_Request_Reply command is used to reply to an LE long term key request event from the controller, and specifies the Long_Term_Key parameter used for this Connection_Handle.

Input parameters

Table 63. **HCI_LE_LONG_TERM_KEY_REQUEST_REPLY input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given	0x0000 ... 0x0EFF
Long_Term_Key	16	128 bit long term key	-

Output parameters

Table 64. **HCI_LE_LONG_TERM_KEY_REQUEST_REPLY output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Connection_Handle	2	Connection handle related to the response	0x0000 ... 0x0EFF

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.39 HCI_LE_LONG_TERM_KEY_REQUESTED_NEGATIVE_REPLY

Description

The LE_Long_Term_Key_Request_Negative_Reply command is used to reply to an LE long term key request event from the controller if the host cannot provide a long term key for this Connection_Handle.

Input parameters

Table 65. **HCI_LE_LONG_TERM_KEY_REQUESTED_NEGATIVE_REPLY input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given	0x0000 ... 0x0EFF

Output parameters

Table 66. **HCI_LE_LONG_TERM_KEY_REQUESTED_NEGATIVE_REPLY output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Connection_Handle	2	Connection handle related to the response	0x0000 ... 0x0EFF

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.40 HCI_LE_READ_SUPPORTED_STATES

Description

The LE_Read_Supported_States command reads the states and state combinations that the link layer supports. LE_States is an 8-octet bit field. If a bit is set to 1 then this state or state combination is supported by the controller. Multiple bits in LE_States may be set to 1 to indicate support for multiple state and state combinations. All the advertising type with the initiate state combinations are set only if the corresponding advertising types and master role combination are set. All the scanning types and the initial state combinations are set only if the corresponding scanning types and master role combination are set.

Input parameters

None

Output parameters

Table 67. **HCI_LE_READ_SUPPORTED_STATES output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
LE_States	8	State or state combination is supported by the controller	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.41 HCI_LE_SET_DATA_LENGTH

Description

The LE_Set_Data_Length command allows the host to suggest maximum transmission packet size and maximum packet transmission time (connMaxTxOctets and connMaxTxTime) to be used for a given connection. The controller may use smaller or larger values based on local information.

Input parameters

Table 68. **HCI_LE_SET_DATA_LENGTH input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
TxOctets	2	Preferred maximum number of payload octets that the local controller includes in a single link layer packet on this connection	0x001B ... 0x00FB
TxTime	2	Preferred maximum number of microseconds that the local controller must use to transmit a single link layer packet on this connection	0x0148 ... 0x4290

Output parameters

Table 69. **HCI_LE_SET_DATA_LENGTH output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.42 HCI_LE_READ_SUGGESTED_DEFAULT_DATA_LENGTH

Description

The LE_Read_Suggested_Default_Data_Length command allows the host to read the host's suggested values (SuggestedMaxTxOctets and SuggestedMaxTxTime) for the controller's maximum transmitted number of payload octets and maximum packet transmission time to be used for new connections.

Input parameters

None

Output parameters

Table 70. **HCI_LE_READ_SUGGESTED_DEFAULT_DATA_LENGTH output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
SuggestedMaxTxOctets	2	The host's suggested value for the controller's maximum transmitted number of payload octets to be used for new connections	0x001B ... 0x00FB
SuggestedMaxTxTime	2	The host's suggested value for the controller's maximum packet transmission time to be used for new connections	0x0148 ... 0x4290

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.43 HCI_LE_WRITE_SUGGESTED_DEFAULT_DATA_LENGTH

Description

The LE_Write_Suggested_Default_Data_Length command allows the host to specify its suggested values for the controller's maximum transmission number of payload octets and maximum packet transmission time to be used for new connections. The controller may use smaller or larger values for connInitialMaxTxOctets and connInitialMaxTxTime based on local information.

Input parameters

Table 71. **HCI_LE_WRITE_SUGGESTED_DEFAULT_DATA_LENGTH input parameters**

Parameter	Size	Description	Possible values
SuggestedMaxTxOctets	2	The host's suggested value for the controller's maximum transmitted number of payload octets to be used for new connections	0x001B ... 0x00FB
SuggestedMaxTxTime	2	The host's suggested value for the controller's maximum packet transmission time to be used for new connections	0x0148 ... 0x4290

Output parameters

Table 72. **HCI_LE_WRITE_SUGGESTED_DEFAULT_DATA_LENGTH output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.44 HCI_LE_READ_LOCAL_P256_PUBLIC_KEY

Description

The LE_Read_Local_P-256_Public_Key command is used to return the local P-256 public key from the controller. The controller generates a new P- 256 public/private key pair upon receipt of this command.

Input parameters

None

Output parameters

Table 73. **HCI_LE_READ_LOCAL_P256_PUBLIC_KEY output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_READ_LOCAL_P256_PUBLIC_KEY_COMPLETE_EVENT

2.1.45 HCI_LE_GENERATE_DHKEY

Description

The LE_Generate_DHKey command is used to initiate generation of a Diffie- Hellman key in the controller for use over the LE transport. This command takes the remote P-256 public key as input. The Diffie-Hellman key generation uses the private key generated by LE_Read_Local_P256_Public_Key command.

Input parameters

Table 74. HCI_LE_GENERATE_DHKEY input parameters

Parameter	Size	Description	Possible values
Remote_P256_Public_Key	64	The remote P-256 public key: X, Y format octets 31-0: X co-ordinate octets 63-32: Y co-ordinate little endian format	-

Output parameters

Table 75. HCI_LE_GENERATE_DHKEY output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_GENERATE_DHKEY_COMPLETE_EVENT

2.1.46 HCI_LE_ADD_DEVICE_TO_RESOLVING_LIST

Description

The LE_Add_Device_To_Resolving_List command is used to add one device to the list of address translations used to resolve resolvable private addresses in the controller.

This command cannot be used when address translation is enabled in the controller and:

- Advertising is enabled
- Scanning is enabled
- Create connection command is outstanding

This command can be used at any time when address translation is disabled in the controller. When a controller cannot add a device to the resolving list because the list is full, it responds with error code 0x07 (memory capacity exceeded).

Input parameters

Table 76. HCI_LE_ADD_DEVICE_TO_RESOLVING_LIST input parameters

Parameter	Size	Description	Possible values
Peer_Identity_Address_Type	1	Identity address type	• 0x00: Public identity address • 0x01: Random (static) identity address
Peer_Identity_Address	6	Public or random (static) identity address of the peer device	-
Peer_IRK	16	IRK of the peer device	-
Local_IRK	16	IRK of the local device	-

Output parameters

Table 77. **HCI_LE_ADD_DEVICE_TO_RESOLVING_LIST output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.47 HCI_LE_REMOVE_DEVICE_FROM_RESOLVING_LIST

Description

The LE_Remove_Device_From_Resolving_List command is used to remove one device from the list of address translations used to resolve resolvable private addresses in the controller.

This command cannot be used when address translation is enabled in the controller and:

- Advertising is enabled
- Scanning is enabled
- Create connection command is outstanding

This command can be used at any time when address translation is disabled in the controller. When a controller cannot remove a device from the resolving list because it is not found, it shall respond with error code 0x02 (unknown connection identifier).

Input parameters

Table 78. **HCI_LE_REMOVE_DEVICE_FROM_RESOLVING_LIST input parameters**

Parameter	Size	Description	Possible values
Peer_Identity_Address_Type	1	Identity address type	• 0x00: Public identity address • 0x01: Random (static) identity address
Peer_Identity_Address	6	Public or random (static) identity address of the peer device	-

Output parameters

Table 79. **HCI_LE_REMOVE_DEVICE_FROM_RESOLVING_LIST output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.48 HCI_LE_CLEAR_RESOLVING_LIST

Description

The LE_Clear_Resolving_List command is used to remove all devices from the list of address translations used to resolve resolvable private addresses in the controller.

This command cannot be used when address translation is enabled in the controller and:

- Advertising is enabled
- Scanning is enabled
- Create connection command is outstanding

This command can be used at any time when address translation is disabled in the controller.

Input parameters

None

Output parameters

Table 80. **HCI_LE_CLEAR_RESOLVING_LIST output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.49 HCI_LE_READ_RESOLVING_LIST_SIZE

Description

The LE_Read_Resolving_List_Size command is used to read the total number of address translation entries in the resolving list that can be stored in the controller.

Input parameters

None

Output parameters

Table 81. **HCI_LE_READ_RESOLVING_LIST_SIZE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Resolving_List_Size	1	Number of address translation entries in the resolving list	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.50 HCI_LE_READ_PEER_RESOLVABLE_ADDRESS

Description

The LE_Read_Peer_Resolvable_Address command is used to get the current peer resolvable private address being used for the corresponding peer public and random (static) identity address. The peer's resolvable address being used may change after the command is called. This command is used at any time. When a controller cannot find a resolvable private address associated with the peer identity address, it responds with error code 0x02 (unknown connection identifier).

Input parameters

Table 82. **HCI_LE_READ_PEER_RESOLVABLE_ADDRESS input parameters**

Parameter	Size	Description	Possible values
Peer_Identity_Address_Type	1	Identity address type	• 0x00: Public identity address • 0x01: Random (static) identity address
Peer_Identity_Address	6	Public or random (static) identity address of the peer device	-

Output parameters

Table 83. **HCI_LE_READ_PEER_RESOLVABLE_ADDRESS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Peer_Resolvable_Address	6	Resolvable private address being used by the peer device	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.51 HCI_LE_READ_LOCAL_RESOLVABLE_ADDRESS

Description

The LE_Read_Local_Resolvable_Address command is used to get the current local resolvable private address being used for the corresponding peer identity address. The local's resolvable address being used may change after the command is called. This command is used at any time. When a controller cannot find a resolvable private address associated with the peer identity address, it responds with error code 0x02 (unknown connection identifier).

Input parameters

Table 84. **HCI_LE_READ_LOCAL_RESOLVABLE_ADDRESS input parameters**

Parameter	Size	Description	Possible values
Peer_Identity_Address_Type	1	Identity address type	• 0x00: Public identity address • 0x01: Random (static) identity address
Peer_Identity_Address	6	Public or random (static) identity address of the peer device	-

Output parameters

Table 85. **HCI_LE_READ_LOCAL_RESOLVABLE_ADDRESS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Local_Resolvable_Address	6	Resolvable private address being used by the local device	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.52 HCI_LE_SET_ADDRESS_RESOLUTION_ENABLE

Description

The LE_Set_Address_Resolution_Enable command is used to enable resolution of resolvable private addresses in the controller. This causes the controller to use the resolving list whenever the controller receives a local or peer resolvable private address. This command can be used at any time except when:

- Advertising is enabled
- Scanning is enabled
- Create connection command is outstanding

Input parameters

Table 86. HCI_LE_SET_ADDRESS_RESOLUTION_ENABLE input parameters

Parameter	Size	Description	Possible values
Address_Resolution_Enable	1	Enable/disable address resolution in the controller. 0x00: Address resolution in controller disabled (default) 0x01: Address resolution in controller enabled	• 0x00: Address resolution in controller disabled (default) • 0x01: Address resolution in controller enabled

Output parameters

Table 87. HCI_LE_SET_ADDRESS_RESOLUTION_ENABLE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.53 HCI_LE_SET_RESOLVABLE_PRIVATE_ADDRESS_TIMEOUT

Description

The LE_Set_Resolvable_Private_Address_Timeout command set the length of time the controller uses a resolvable private address before a new resolvable private address is generated and starts being used. This timeout applies to all addresses generated by the controller.

Input parameters

Table 88. HCI_LE_SET_RESOLVABLE_PRIVATE_ADDRESS_TIMEOUT input parameters

Parameter	Size	Description	Possible values
RPA_Timeout	2	RPA_Timeout measured in seconds. Range for N: 0x0001 - 0xA1B8 (1 s - approximately 11.5 hours) Default: N= 0x0384 (900 s or 15 minutes)	-

Output parameters

Table 89. HCI_LE_SET_RESOLVABLE_PRIVATE_ADDRESS_TIMEOUT output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.54 HCI_LE_READ_MAXIMUM_DATA_LENGTH

Description

The LE_Read_Maximum_Data_Length command allows the host to read the controller maximum supported payload octets and packet duration times for transmission and reception (supportedMaxTxOctets and supportedMaxTxTime, supportedMaxRxOctets, and supportedMaxRxTime).

Input parameters

None

Output parameters

Table 90. HCI_LE_READ_MAXIMUM_DATA_LENGTH output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
supportedMaxTxOctets	2	Maximum number of payload octets that the local controller supports for transmission of a single link layer packet on a data connection	0x001B ... 0x00FB
supportedMaxTxTime	2	Maximum time, in microseconds, that the local controller supports for transmission of a single link layer packet on a data connection	0x0148 ... 0x4290
supportedMaxRxOctets	2	Maximum number of payload octets that the local controller supports for reception of a single link layer packet on a data connection	0x001B ... 0x00FB
supportedMaxRxTime	2	Maximum time, in microseconds, that the local controller supports for reception of a single link layer packet on a data connection	0x0148 ... 0x4290

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.55 HCI_LE_READ_PHY

Description

1.1.55.1 Description

The LE_Read_PHY command is used to read the current transmitter PHY and receiver PHY on the connection identified by the Connection_Handle.

Input parameters

Table 91. HCI_LE_READ_PHY input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given	0x0000 ... 0x0EFF

Output parameters

Table 92. **HCI_LE_READ_PHY output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Connection_Handle	2	Connection handle related to the response	0x0000 ... 0x0EFF
TX_PHY	1	Transmitter PHY in use	• 0x01: The transmitter PHY for the connection is LE 1M • 0x02: The transmitter PHY for the connection is LE 2M • 0x03: The transmitter PHY for the connection is LE coded (not supported by STM32WB)
RX_PHY	1	Receiver PHY in use	• 0x01: The receiver PHY for the connection is LE 1M • 0x02: The receiver PHY for the connection is LE 2M • 0x03: The receiver PHY for the connection is LE coded (not supported by STM32WB)

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.56 HCI_LE_SET_DEFAULT_PHY

Description

The LE_Set_Default_PHY command allows the host to specify its preferred values for the transmitter PHY and receiver PHY to be used for all subsequent connections over the LE transport. The ALL_PHYS parameter is a bit field that allows the host to specify, for each direction, whether it has no preference among the PHYs that the controller supports in a given direction or whether it has specified particular PHYs that it prefers in the TX_PHYS or RX_PHYS parameter. The TX_PHYS parameter is a bit field that indicates the transmitter PHYs that the Host prefers the controller to use. If the ALL_PHYS parameter specifies that the host has no preference, the TX_PHYS parameter is ignored; otherwise at least one bit is set to 1. The RX_PHYS parameter is a bit field that indicates the receiver PHYs that the host prefers the controller to use. If the ALL_PHYS parameter specifies that the host has no preference, the RX_PHYS parameter is ignored; otherwise at least one bit is set to 1.

Input parameters

Table 93. **HCI_LE_SET_DEFAULT_PHY input parameters**

Parameter	Size	Description	Possible values
ALL_PHYS	1	Host preferences for TX PHY and RX PHY	0x00 ... 0x03
TX_PHYS	1	Host preferences for TX PHY (no LE coded support)	0x00 ... 0x03
RX_PHYS	1	Host preferences for RX PHY (no LE coded support)	0x00 ... 0x03

Output parameters

Table 94. **HCI_LE_SET_DEFAULT_PHY output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.57 HCI_LE_SET_PHY

Description

The LE_Set_PHY command is used to set the PHY preferences for the connection identified by the Connection_Handle. The controller might not be able to make the change (e.g. because the peer does not support the requested PHY) or may decide that the current PHY is preferable. The ALL_PHYS parameter is a bit field that allows the host to specify, for each direction, whether it has no preference among the PHYs that the controller supports in a given direction or whether it has specified particular PHYs that it prefers in the TX_PHYS or RX_PHYS parameter.

The TX_PHYS parameter is a bit field that indicates the transmitter PHYs that the Host prefers the controller to use. If the ALL_PHYS parameter specifies that the host has no preference, the TX_PHYS parameter is ignored; otherwise at least one bit is set to 1. The RX_PHYS parameter is a bit field that indicates the receiver PHYs that the Host prefers the controller to use. If the ALL_PHYS parameter specifies that the host has no preference, the RX_PHYS parameter is ignored; otherwise at least one bit is set to 1. If, for at least one direction, the host has specified a preference and the current PHY is not one of those preferred, the controller requests a change. Otherwise the controller may, but need not, request a change. The PHY preferences provided by the LE Set PHY command override those provided via the LE set default PHY command or any preferences previously set using the LE set PHY command on the same connection. The PHY_options parameter is a bit field that allows the host to specify options for PHYs. The default value for a new connection is all zero bits. The controller may override any preferred coding for transmitting on the LE coded PHY. The host may specify a preferred coding even if it prefers not to use the LE coded transmitter PHY since the controller may override the PHY preference.

Input parameters

Table 95. HCI_LE_SET_PHY input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given	0x0000 ... 0x0EFF
ALL_PHYS	1	Host preferences for TX PHY and RX PHY	0x00 ... 0x03
TX_PHYS	1	Host preferences for TX PHY (no LE coded support)	0x00 ... 0x03
RX_PHYS	1	Host preferences for RX PHY (no LE coded support)	0x00 ... 0x03
PHY_options	2	Not supported by STM32WB	0x00 ... 0x03

Output parameters

Table 96. HCI_LE_SET_PHY output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT

2.2 HCI testing commands

In Table 97 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / SO, respectively, for Link layer only / Beacon only / Slave only.

Table 97. HCI testing commands list

Command	OpCode	LO	SO	BO
HCI_LE_RECEIVER_TEST	0x201D	Y	Y	-
HCI_LE_TRANSMITTER_TEST	0x201E	Y	Y	-
HCI_LE_TEST_END	0x201F	Y	Y	-
HCI_LE_ENHANCED_RECEIVER_TEST	0x2033	Y	-	-
HCI_LE_ENHANCED_TRANSMITTER_TEST	0x2034	Y	-	-

2.2.1 HCI_LE_RECEIVER_TEST

Description

This command is used to start a test where the DUT receives test reference packets at a fixed interval. The tester generates the test reference packets.

Input parameters

Table 98. HCI_LE_RECEIVER_TEST input parameters

Parameter	Size	Description	Possible values
RX_Frequency	1	N = (F - 2402) / 2 frequency range : 2402 MHz to 2480 MHz	0x00 ... 0x27

Output parameters

Table 99. HCI_LE_RECEIVER_TEST output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.2.2 HCI_LE_TRANSMITTER_TEST

Description

This command is used to start a test where the DUT generates test reference packets at a fixed interval. The controller transmits at maximum power. An LE controller supporting the LE_Transmitter_Test command supports Packet_Payload values 0x00, 0x01 and 0x02. An LE controller supports other values of Packet_Payload.

Input parameters

Table 100. **HCI_LE_TRANSMITTER_TEST input parameters**

Parameter	Size	Description	Possible values
TX_Frequency	1	N = (F - 2402) / 2 Frequency range : 2402 MHz to 2480 MHz	0x00 ... 0x27
Length_Of_Test_Data	1	Length in bytes of payload data in each packet.	0x00 ... 0x25
Packet_Payload	1	Type of packet payload.	• 0x00: Pseudo-Random bit sequence 9 • 0x01: Pattern of alternating bits '11110000' • 0x02: Pattern of alternating bits '10101010' • 0x03: Pseudo-Random bit sequence 15 • 0x04: Pattern of All '1' bits • 0x05: Pattern of All '0' bits • 0x06: Pattern of alternating bits '00001111' • 0x07: Pattern of alternating bits '0101'

Output parameters

Table 101. **HCI_LE_TRANSMITTER_TEST output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_DISCONNECTION_COMPLETE_EVENT

2.2.3 HCI_LE_TEST_END

Description

This command is used to stop any test which is in progress. The Number_Of_Packets for a transmitter test is reported as 0x0000. The Number_Of_Packets is an unsigned number and contains the number of received packets.

Input parameters

None

Output parameters

Table 102. **HCI_LE_TEST_END output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Number_Of_Packets	2	Number of packets received	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.2.4 HCI_LE_ENHANCED_RECEIVER_TEST

Description

This command is used to start a test where the DUT receives test reference packets at a fixed interval. The tester generates the test reference packets.

Input parameters

Table 103. **HCI_LE_ENHANCED_RECEIVER_TEST input parameters**

Parameter	Size	Description	Possible values
RX_Frequency	1	N = (F - 2402) / 2 Frequency Range : 2402 MHz to 2480 MHz	0x00 ... 0x27
PHY	1	PHY to use for test packet	• 0x00: Reserved for future use • 0x01: Transmitter set to use the LE 1M PHY • 0x02: Transmitter set to use the LE 2M PHY • 0x03: Transmitter set to use the LE coded PHY with S=8 data coding • 0x04: Transmitter set to use the LE coded PHY with S=2 data coding
Modulation_Index	1	Modulation index capability of the transmitter	• 0x00: Assume transmitter has a standard modulation index • 0x01: Assume transmitter has a stable modulation index

Output parameters

Table 104. **HCI_LE_ENHANCED_RECEIVER_TEST output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.2.5 HCI_LE_ENHANCED_TRANSMITTER_TEST

Description

This command is used to start a test where the DUT generates test reference packets at a fixed interval. The controller transmits at maximum power. An LE controller supporting the LE_Enhanced Transmitter_Test command supports Packet_Payload values 0x00, 0x01 and 0x02. An LE controller supporting the LE coded PHY also supports Packet_Payload value 0x04 (not supported by STM32WB). An LE controller may support other values of Packet_Payload.

Input parameters

Table 105. HCI_LE_ENHANCED_TRANSMITTER_TEST input parameters

Parameter	Size	Description	Possible values
TX_Frequency	1	N = (F - 2402) / 2 Frequency range : 2402 MHz to 2480 MHz	0x00 ... 0x27
Length_Of_Test_Data	1	Length in bytes of payload data in each packet.	0x00 ... 0x25
Packet_Payload	1	Type of packet payload.	• 0x00: Pseudo-random bit sequence 9 • 0x01: Pattern of alternating bits '11110000' • 0x02: Pattern of alternating bits '10101010' • 0x03: Pseudo-random bit sequence 15 • 0x04: Pattern of All '1' bits • 0x05: Pattern of All '0' bits • 0x06: Pattern of alternating bits '00001111' • 0x07: Pattern of alternating bits '0101'
PHY	1	PHY to use for test packet	• 0x00: Reserved for future use • 0x01: Transmitter set to use the LE 1M PHY • 0x02: Transmitter set to use the LE 2M PHY • 0x03: Transmitter set to use the LE coded PHY with S=8 data coding • 0x04: Transmitter set to use the LE coded PHY with S=2 data coding

Output parameters

Table 106. HCI_LE_ENHANCED_TRANSMITTER_TEST output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3 HAL commands

In Table 107 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / SO, respectively, for Link layer only / Beacon only / Slave only.

Table 107. HAL commands list

Command	OpCode	LO	SO	BO
ACI_HAL_GET_FW_BUILD_NUMBER	0xFC00	Y	Y	Y
ACI_HAL_WRITE_CONFIG_DATA	0xFC0C	Y	Y	Y
ACI_HAL_READ_CONFIG_DATA	0xFC0D	Y	Y	Y
ACI_HAL_SET_TX_POWER_LEVEL	0xFC0F	Y	Y	Y
ACI_HAL_LE_TX_TEST_PACKET_NUMBER	0xFC14	Y	Y	-
ACI_HAL_TONE_START	0xFC15	Y	Y	-
ACI_HAL_TONE_STOP	0xFC16	Y	Y	-
ACI_HAL_GET_LINK_STATUS	0xFC17	Y	-	-
ACI_HAL_SET_RADIO_ACTIVITY_MASK	0xFC18	Y	Y	Y
ACI_HAL_GET_ANCHOR_PERIOD	0xFC19	Y	-	-
ACI_HAL_SET_EVENT_MASK	0xFC1A	-	-	-
ACI_HAL_GET_PM_DEBUG_INFO	0xFC1C	Y	-	-
ACI_HAL_READ_RADIO_REG	0xFC30	Y	Y	-
ACI_HAL_WRITE_RADIO_REG	0xFC31	Y	Y	-
ACI_HAL_READ_RAW_RSSI	0xFC32	Y	Y	-
ACI_HAL_RX_START	0xFC33	Y	Y	-
ACI_HAL_RX_STOP	0xFC34	Y	Y	-
ACI_HAL_STACK_RESET	0xFC3B	Y	Y	Y

2.3.1 ACI_HAL_GET_FW_BUILD_NUMBER

Description

This command returns the build number associated with the firmware version currently running.

Input parameters

None

Output parameters

Table 108. ACI_HAL_GET_FW_BUILD_NUMBER output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Build_Number	2	Build number of the firmware.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.2 ACI_HAL_WRITE_CONFIG_DATA

Description

This command writes a value to a low level configure data structure. It is useful to setup directly some low level parameters for the system in the runtime.

Input parameters

Table 109. ACI_HAL_WRITE_CONFIG_DATA input parameters

Parameter	Size	Description	Possible values
Offset	1	Offset of the element in the configuration data structure which has to be written. The valid offsets are: • 0x00: Bluetooth public address, value length to be written: 6 bytes • 0x08: Encryption root key used to derive LTK and CSRK, value length to be written: 16 bytes • 0x18: Identity root key used to derive LTK and CSRK, value length to be written: 16 bytes • 0x2E: Static random address: 6 bytes	• 0x00: CONFIG_DATA_PUBADDR_OFFSET • 0x08: CONFIG_DATA_ER_OFFSET • 0x18: CONFIG_DATA_IR_OFFSET • 0x2E: CONFIG_DATA_RANDOM_ADDRESS_WR
Length	1	Length of data to be written	-
Value	Length	Data to be written	-

Output parameters

Table 110. ACI_HAL_WRITE_CONFIG_DATA output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.3 ACI_HAL_READ_CONFIG_DATA

Description

This command requests the value in the low level configure data structure. The number of read bytes changes for different offset.

Input parameters

Table 111. **ACI_HAL_READ_CONFIG_DATA input parameters**

Parameter	Size	Description	Possible values
Offset	1	Offset of the element in the configuration data structure which has to be read. The valid offsets are: • 0x00: Bluetooth public address, value length returned: 6 bytes • 0x08: Encryption root key used to derive LTK and CSRK, value length returned: 16 bytes • 0x18: Identity root key used to derive LTK and CSRK, value length returned: 16 bytes • 0x80: Static random address. Value length returned: 6 bytes (read-only)	• 0x00: CONFIG_DATA_PUBADDR_OFFSET • 0x08: CONFIG_DATA_ER_OFFSET • 0x18: CONFIG_DATA_IR_OFFSET • 0x80: CONFIG_DATA_RANDOM_ADDRESS

Output parameters

Table 112. **ACI_HAL_READ_CONFIG_DATA output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Data_Length	1	Length of Data in octets	-
Data	Data_Length	Data field associated with Offset parameter	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.4 ACI_HAL_SET_TX_POWER_LEVEL

Description

This command sets the TX power level of the device. By controlling the PA_LEVEL, that determines the output power level (dBm) at the IC pin. When the system starts up or reboots, the default TX power level (the maximum value of 6 dBm) is used. Once this command is given, the output power is changed instantaneously, regardless if there is an ongoing Bluetooth communication or not. For example, for debugging purpose, the device can be set to advertise all the time. a nd use this command to observe the change of signal strength. The system keeps the last received TX power level from the command, i.e. the seconnd command overwrites the previous TX power level. The new TX power level remains until another Set TX power command, or the system reboots.

Input parameters

Table 113. **ACI_HAL_SET_TX_POWER_LEVEL input parameters**

Parameter	Size	Description	Possible values
En_High_Power	1	Enable high power mode - deprecated and ignored on STM32WB	• 0x00: Standard power • 0x01: High power
PA_Level	1	Power amplifier output level. Output power is indicative and it depends on the PCB layout and associated components. Here the values are given at the IC pin	• 0x00: -40 dBm • 0x01: -20.85 dBm • 0x02: -19.75 dBm • 0x03: -18.85 dBm • 0x04: -17.6 dBm • 0x05: -16.5 dBm • 0x06: -15.25 dBm • 0x07: -14.1 dBm • 0x08: -13.15 dBm • 0x09: -12.05 dBm • 0x0A: -10.9 dBm • 0x0B: -9.9 dBm • 0x0C: -8.85 dBm • 0x0D: -7.8 dBm • 0x0E: -6.9 dBm • 0x0F: -5.9 dBm • 0x10: -4.95 dBm • 0x11: -4 dBm • 0x12: -3.15 dBm • 0x13: -2.45 dBm • 0x14: -1.8 dBm • 0x15: -1.3 dBm • 0x16: -0.85 dBm • 0x17: -0.5 dBm • 0x18: -0.15 dBm • 0x19: 0 dBm • 0x1A: +1 dBm • 0x1B: +2 dBm • 0x1C: +3 dBm • 0x1D: +4 dBm • 0x1E: +5 dBm • 0x1F: +6 dBm

Output parameters

Table 114. **ACI_HAL_SET_TX_POWER_LEVEL output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.5 ACI_HAL_LE_TX_TEST_PACKET_NUMBER

Description

This command returns the number of packets sent in direct test mode. When the direct TX test is started, a 32-bit counter is used to count how many packets have been transmitted. This command can be used to check how many packets have been sent during the direct TX test. The counter starts from zero and counts upwards. The counter can wrap and start from zero again. The counter is not cleared until the next direct TX test starts.

Input parameters

None

Output parameters

Table 115. ACI_HAL_LE_TX_TEST_PACKET_NUMBER output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Number_Of_Packets	4	Number of packets sent during the last direct TX test.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.6 ACI_HAL_TONE_START

Description

This command starts a carrier frequency, i.e. a tone, on a specific channel. The frequency sine wave at the specific channel may be used for debugging purpose only. The channel ID is a parameter from 0x00 to 0x27 for the 40 BLE channels, e.g. 0x00 for 2.402 GHz, 0x01 for 2.404 GHz etc. This command must not be used when normal Bluetooth activities are ongoing. The tone must be stopped by ACI_HAL_TONE_STOP command.

Input parameters

Table 116. ACI_HAL_TONE_START input parameters

Parameter	Size	Description	Possible values
RF_Channel	1	BLE Channel ID, from 0x00 to 0x27 meaning (2.402 + 2*0xXX) GHz device continuously emits 0 s, that means that the tone is at the channel center frequency less the maximum frequency deviation (250 kHz).	0x00 ... 0x27
Freq_offset	1	Frequency offset for tone channel	0x00 ... 0xFF

Output parameters

Table 117. ACI_HAL_TONE_START output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.7 ACI_HAL_TONE_STOP

Description

This command is used to stop the previously started ACI_HAL_TONE_START command.

Input parameters

None

Output parameters

Table 118. **ACI_HAL_TONE_STOP output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.8 ACI_HAL_GET_LINK_STATUS

Description

This command returns the status of the eight Bluetooth Low Energy links managed by the device.

Input parameters

None

Output parameters

Table 119. **ACI_HAL_GET_LINK_STATUS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Link_Status	8	Array of link status (eight links). Each link status is 1 byte.	• 0x00: Idle • 0x01: Advertising • 0x02: Connected in slave role • 0x03: Scanning • 0x04: Reserved • 0x05: Connected in master role • 0x06: TX test mode • 0x07: RX test mode
Link_Connection_Handle	16	Array of connection handles (two bytes) for eight links.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.9 ACI_HAL_SET_RADIO_ACTIVITY_MASK

Description

This command set the bitmask associated to ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT. Only the radio activities enabled in the mask is reported to application by ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT.

Input parameters

Table 120. **ACI_HAL_SET_RADIO_ACTIVITY_MASK input parameters**

Parameter	Size	Description	Possible values
Radio_Activity_Mask	2	Bitmask of radio events	Bitmask of: • 0x0001: Idle • 0x0002: Advertising • 0x0004: Connection event slave • 0x0008: Scanning • 0x0010: Connection request • 0x0020: Connection event master • 0x0040: TX test mode • 0x0080: RX test mode

Output parameters

Table 121. **ACI_HAL_SET_RADIO_ACTIVITY_MASK output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT

2.3.10 ACI_HAL_GET_ANCHOR_PERIOD

Description

This command returns information about the Anchor Period to help application in selecting slot timings when operating in multi-link scenarios.

Input parameters

None

Output parameters

Table 122. **ACI_HAL_GET_ANCHOR_PERIOD output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Anchor_Period	4	Current anchor period. T = N * 0.625 ms	-
Max_Free_Slot	4	Maximum available time that can be allocated for a new slot. T = N * 0.625 ms	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.11 ACI_HAL_SET_EVENT_MASK

Description

This command is used to enable/disable the generation of HAL events.

Input parameters

Table 123. **ACI_HAL_SET_EVENT_MASK input parameters**

Parameter	Size	Description	Possible values
Event_Mask	4	Mask to enable/disable generation of HAL events	Bitmask of: • 0x00000000: No events specified (Default) • 0x00000001: ACI_HAL_SCAN_REQ_REPORT_EVENT

Output parameters

Table 124. **ACI_HAL_SET_EVENT_MASK output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.12 ACI_HAL_GET_PM_DEBUG_INFO

Description

This command is used to retrieve TX, RX and total buffer count allocated for ACL packets.

Input parameters

None

Output parameters

Table 125. **ACI_HAL_GET_PM_DEBUG_INFO output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Allocated_For_TX	1	MBlocks allocated for TXing	-
Allocated_For_RX	1	MBlocks allocated for RXing	-
Allocated_MBlocks	1	Overall allocated MBlocks	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.13 ACI_HAL_READ_RADIO_REG

Description

This command Reads Register value from the RF module

Input parameters

Table 126. **ACI_HAL_READ_RADIO_REG input parameters**

Parameter	Size	Description	Possible values
Register_Address	1	Address of the register to be read	-

Output parameters

Table 127. **ACI_HAL_READ_RADIO_REG output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
reg_val	1	Register value	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.14 ACI_HAL_WRITE_RADIO_REG

Description

This command writes Register value to the RF module

Input parameters

Table 128. **ACI_HAL_WRITE_RADIO_REG input parameters**

Parameter	Size	Description	Possible values
Register_Address	1	Address of the register to be written	-
Register_Value	1	Value to be written	-

Output parameters

Table 129. **ACI_HAL_WRITE_RADIO_REG output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.15 ACI_HAL_READ_RAW_RSSI

Description

This command returns the raw value of the RSSI

Input parameters

None

Output parameters

Table 130. **ACI_HAL_READ_RAW_RSSI output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.16 ACI_HAL_RX_START

Description

This command does set up the RF to listen to a specific RF channel

Input parameters

Table 131. **ACI_HAL_RX_START input parameters**

Parameter	Size	Description	Possible values
RF_Channel	1	BLE channel ID, from 0x00 to 0x27 meaning (2.402 + 2*0xXX) GHz device continuously emits 0s, that means that the tone is at the channel centre frequency less the maximum frequency deviation (250kHz).	0x00 ... 0x27

Output parameters

Table 132. **ACI_HAL_RX_START output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.17 ACI_HAL_RX_STOP

Description

This command stop a previous ACI_HAL_RX_START command

Input parameters

None

Output parameters

Table 133. **ACI_HAL_RX_STOP output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.18 ACI_HAL_STACK_RESET

Description

This command is equivalent to HCI_RESET but ensures the sleep mode is entered immediately after its completion.

Input parameters

None

Output parameters

Table 134. **ACI_HAL_STACK_RESET output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4 GAP commands

In Table 135 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / SO, respectively, for Link layer only / Beacon only / Slave only.

Table 135. GAP commands list

Command	OpCode	LO	SO	BO
ACI_GAP_SET_NON_DISCOVERABLE	0xFC81	-	Y	-
ACI_GAP_SET_LIMITED_DISCOVERABLE	0xFC82	-	Y	-
ACI_GAP_SET_DISCOVERABLE	0xFC83	-	Y	-
ACI_GAP_SET_DIRECT_CONNECTABLE	0xFC84	-	Y	-
ACI_GAP_SET_IO_CAPABILITY	0xFC85	-	Y	-
ACI_GAP_SET_AUTHENTICATION_REQUIREMENT	0xFC86	-	Y	-
ACI_GAP_SET_AUTHORIZATION_REQUIREMENT	0xFC87	-	Y	-
ACI_GAP_PASS_KEY_RESP	0xFC88	-	Y	-
ACI_GAP_AUTHORIZATION_RESP	0xFC89	-	Y	-
ACI_GAP_INIT	0xFC8A	-	Y	-
ACI_GAP_SET_NON_CONNECTABLE	0xFC8B	-	Y	-
ACI_GAP_SET_UNDIRECTED_CONNECTABLE	0xFC8C	-	Y	-
ACI_GAP_SLAVE_SECURITY_REQ	0xFC8D	-	Y	-
ACI_GAP_UPDATE_ADV_DATA	0xFC8E	-	Y	-
ACI_GAP_DELETE_AD_TYPE	0xFC8F	-	Y	-
ACI_GAP_GET_SECURITY_LEVEL	0xFC90	-	Y	-
ACI_GAP_SET_EVENT_MASK	0xFC91	-	Y	-
ACI_GAP_CONFIGURE_WHITELIST	0xFC92	-	Y	-
ACI_GAP_TERMINATE	0xFC93	-	Y	-
ACI_GAP_CLEAR_SECURITY_DB	0xFC94	-	Y	-
ACI_GAP_ALLOW_REBOND	0xFC95	-	Y	-
ACI_GAP_START_LIMITED_DISCOVERY_PROC	0xFC96	-	-	-
ACI_GAP_START_GENERAL_DISCOVERY_PROC	0xFC97	-	-	-
ACI_GAP_START_NAME_DISCOVERY_PROC	0xFC98	-	-	-
ACI_GAP_START_AUTO_CONNECTION_ESTABLISH_PROC	0xFC99	-	-	-
ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC	0xFC9A	-	-	-
ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC	0xFC9B	-	-	-
ACI_GAP_CREATE_CONNECTION	0xFC9C	-	-	-
ACI_GAP_TERMINATE_GAP_PROC	0xFC9D	-	-	-
ACI_GAP_START_CONNECTION_UPDATE	0xFC9E	-	-	-
ACI_GAP_SEND_PAIRING_REQ	0xFC9F	-	-	-
ACI_GAP_RESOLVE_PRIVATE_ADDR	0XFCA0	-	-	-
ACI_GAP_SET_BROADCAST_MODE	0xFCA1	-	-	-
ACI_GAP_START_OBSERVATION_PROC	0xFCA2	-	-	-
ACI_GAP_GET_BONDED_DEVICES	0xFCA3	-	Y	-
ACI_GAP_IS_DEVICE_BONDED	0xFCA4	-	Y	-

Command	OpCode	LO	SO	BO
ACI_GAP_NUMERIC_COMPARISON_VALUE_CONFIRM_YESNO	0xFCA5	-	Y	-
ACI_GAP_PASSKEY_INPUT	0xFCA6	-	Y	-
ACI_GAP_GET_OOB_DATA	0xFCA7	-	Y	-
ACI_GAP_SET_OOB_DATA	0xFCA8	-	Y	-
ACI_GAP_ADD_DEVICES_TO_RESOLVING_LIST	0xFCA9	-	Y	-
ACI_GAP_REMOVE_BONDED_DEVICE	0xFCAA	-	Y	-

2.4.1 ACI_GAP_SET_NON_DISCOVERABLE

Description

Put the device in non-discoverable mode. This command disables the LL advertising.

Input parameters

None

Output parameters

Table 136. ACI_GAP_SET_NON_DISCOVERABLE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.2 ACI_GAP_SET_LIMITED_DISCOVERABLE

Description

Put the device in limited discoverable mode. The device is discoverable for maximum period of TGAP (lim_adv_timeout) = 180 seconds (from errata). The advertising can be disabled at any time by issuing ACI_GAP_SET_NON_DISCOVERABLE command. The Adv_Interval_Min and Adv_Interval_Max parameters are optional. If both are set to zero, the GAP uses default values for adv intervals for limited discoverable mode (250 ms and 500 ms respectively). To allow a fast connection, the host can set Local_Name, Service_Uuid_List, Slave_Conn_Interval_Min and Slave_Conn_Interval_Max. If provided, these data are inserted into the advertising packet payload as AD data. These parameters are optional in this command. These values can be set in advertised data using GAP_Update_Adv_Data command separately. The total size of data in advertising packet cannot exceed 31 bytes. With this command, the BLE stack also adds automatically the following standard AD types:

- AD Flags
- Power level when advertising timeout happens (i.e. limited discovery period has elapsed), controller generates ACI_GAP_LIMITED_DISCOVERABLE_EVENT event

Input parameters

Table 137. ACI_GAP_SET_LIMITED_DISCOVERABLE input parameters

Parameter	Size	Description	Possible values
Advertising_Type	1	Advertising type. Advertising_Type type cannot be any of GAP_ADV_HIGH_DC_DIRECT_IND or GAP_ADV_HIGH_DC_DIRECT_IND.	• 0x00: ADV_IND (connectable undirected advertising) • 0x01: ADV_DIRECT_IND (connectable directed advertising)

Parameter	Size	Description	Possible values
			• 0x02: ADV_SCAN_IND (scannable undirected advertising) • 0x03: ADV_NONCONN_IND (non connectable undirected advertising)
Advertising_Interval_Min	2	Minimum advertising interval. Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address
Advertising_Filter_Policy	1	Advertising filter policy: not applicable (the value of Advertising_Filter_Policy parameter is not used inside the Stack)	-
Local_Name_Length	1	Length of the local_name field in octets. If length is set to 0x00, Local_Name parameter is not used.	-
Local_Name	Local_Name_Length	Local name of the device. First byte must be 0x08 for shortened local name or 0x09 for complete local name. No NULL character at the end.	-
Service_Uuid_length	1	Length of the service uuid list in octets. If there is no service to be advertised, set this field to 0x00.	-
Service_Uuid_List	Service_Uuid_length	This is the list of the uuids. First byte is the AD Type.	-
Slave_Conn_Interval_Min	2	Slave connection interval minimum value suggested by peripheral. If Slave_Conn_Interval_Min and Slave_Conn_Interval_Max are not 0x0000, slave connection interval range AD structure is added in advertising data. Connection interval is defined in the following manner: connIntervalmin = Slave_Conn_Interval_Min x 1.25ms.	• 0x0000 (NaN) • 0xFFFF (NaN) : No specific minimum • 0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Slave_Conn_Interval_Max	2	Slave connection interval maximum value suggested by peripheral. If Slave_Conn_Interval_Min and Slave_Conn_Interval_Max are not 0x0000, slave connection interval range AD structure is added in advertising data. Connection interval is defined in the following manner: connIntervalmax = Slave_Conn_Interval_Max x 1.25ms	• 0x0000 (NaN) • 0xFFFF (NaN) : No specific maximum • 0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)

Output parameters

Table 138. **ACI_GAP_SET_LIMITED_DISCOVERABLE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- ACI_GAP_LIMITED_DISCOVERABLE_EVENT

2.4.3 ACI_GAP_SET_DISCOVERABLE

Description

Put the device in general discoverable mode. The device is discoverable until the host issues the ACI_GAP_SET_NON_DISCOVERABLE command. The Adv_Interval_Min and Adv_Interval_Max parameters are optional. If both are set to 0, the GAP uses the default values for adv intervals for general discoverable mode. When using connectable undirected advertising events:

- Adv_Interval_Min = 30 ms
- Adv_Interval_Max = 60 ms

When using non-connectable advertising events or scannable undirected advertising events:

- Adv_Interval_Min = 100 ms
- Adv_Interval_Max = 150 ms

Host can set the local name, a service UUID list and the slave connection interval range. If provided, these data are inserted into the advertising packet payload as AD data. These parameters are optional in this command. These values can be also set using aci_gap_update_adv_data() separately. The total size of data in advertising packet cannot exceed 31 bytes. With this command, the BLE stack also adds automatically the following standard AD types:

- AD Flags
- TX power level

Input parameters

Table 139. **ACI_GAP_SET_DISCOVERABLE input parameters**

Parameter	Size	Description	Possible values
Advertising_Type	1	Advertising type. Advertising_Type type cannot be any of GAP_ADV_HIGH_DC_DIRECT_IND or GAP_ADV_HIGH_DC_DIRECT_IND.	• 0x00: ADV_IND (Connectable undirected advertising) • 0x01: ADV_DIRECT_IND (Connectable directed advertising) • 0x02: ADV_SCAN_IND (Scannable undirected advertising) • 0x03: ADV_NONCONN_IND (Non connectable undirected advertising)
Advertising_Interval_Min	2	Minimum advertising interval. Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Advertising_Interval_Max	2	Maximum advertising interval Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled)	• 0x00: Public device address • 0x01: Random device address

Parameter	Size	Description	Possible values
		• 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x02: Resolvable private address • 0x03: Non resolvable private address
Advertising_Filter_Policy	1	Advertising filter policy: not applicable (the value of Advertising_Filter_Policy parameter is not used inside the stack)	-
Local_Name_Length	1	Length of the local_name field in octets. If length is set to 0x00, Local_Name parameter is not used.	-
Local_Name	Local_Name_Length	Local name of the device. First byte must be 0x08 for shortened local name or 0x09 for complete local name. No NULL character at the end.	-
Service_Uuid_length	1	Length of the service UUID list in octets. If there is no service to be advertised, set this field to 0x00.	-
Service_Uuid_List	Service_Uuid_length	This is the list of the UUIDs . First byte is the AD type.	-
Slave_Conn_Interval_Min	2	Slave connection interval minimum value suggested by peripheral. If Slave_Conn_Interval_Min and Slave_Conn_Interval_Max are not 0x0000, slave connection interval range AD structure is added in advertising data. Connection interval is defined in the following manner: connIntervalmin = Slave_Conn_Interval_Min x 1.25ms.	• 0x0000 (NaN) • 0xFFFF (NaN) : No specific minimum • 0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Slave_Conn_Interval_Max	2	Slave connection interval maximum value suggested by peripheral. If Slave_Conn_Interval_Min and Slave_Conn_Interval_Max are not 0x0000, slave connection interval range AD structure is added in advertising data. Connection interval is defined in the following manner: connIntervalmax = Slave_Conn_Interval_Max x 1.25ms	• 0x0000 (NaN) • 0xFFFF (NaN) : No specific maximum • 0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)

Output parameters

Table 140. ACI_GAP_SET_DISCOVERABLE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.4 ACI_GAP_SET_DIRECT_CONNECTABLE

Description

Set the device in direct connectable mode. Device uses direct connectable mode to advertise using high duty cycle advertisement events or low duty cycle advertisement events and the address as either what is specified in the own address type parameter. The command specifies the type of the advertising used. If the privacy is enabled, the type parameter in reconnection address is used for advertising, otherwise the address of the type specified in OwnAddrType is used. The device is in directed connectable mode only for 1.28 seconds. If no connection is established within this duration, the device enters non discoverable mode and advertising must be again enabled explicitly. The controller generates a HCI_ADVERTISING_TIMEOUT_ERR_CODE if the connection was not established and BLE_STATUS_SUCCESS (0x00) if the connection was successfully established. If host privacy is enabled this command returns BLE_STATUS_INVALID_PARAMS.

Input parameters

Table 141. ACI_GAP_SET_DIRECT_CONNECTABLE input parameters

Parameter	Size	Description	Possible values
Own_Address_Type	1	Own address type: • 0x00: Public device address (only if privacy is disabled) • 0x01: Random device address (only if privacy is disabled) • 0x02: Resolvable private address (only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address
Directed_Advertising_Type	1	Type of directed advertising.	• 0x01: High duty cycle directed advertising • 0x04: Low duty cycle directed advertising
Direct_Address_Type	1	The address type of the peer device.	• 0x00: Public device address • 0x01: Random device address
Direct_Address	6	Initiator Bluetooth address	-
Advertising_Interval_Min	2	Minimum advertising interval. Time = N * 0.625 ms	• 0x0006 (3.750 ms): for high duty cycle directed advertising • 0x0020 (20.000 ms) ... 0x4000 (10240.000 ms): for low duty cycle directed advertising
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 msec.	• 0x0006 (3.750 ms) : for high duty cycle directed advertising • 0x0020 (20.000 ms) ... 0x4000 (10240.000 ms) : for low duty cycle directed advertising

Output parameters

Table 142. ACI_GAP_SET_DIRECT_CONNECTABLE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- HCI_LE_CONNECTION_COMPLETE_EVENT

2.4.5 ACI_GAP_SET_IO_CAPABILITY

Description

Set the IO capabilities of the device. This command has to be given only when the device is not in a connected state.

Input parameters

Table 143. **ACI_GAP_SET_IO_CAPABILITY input parameters**

Parameter	Size	Description	Possible values
IO_Capability	1	IO capability of the device.	• 0x00: IO_CAP_DISPLAY_ONLY • 0x01: IO_CAP_DISPLAY_YES_NO • 0x02: IO_CAP_KEYBOARD_ONLY • 0x03: IO_CAP_NO_INPUT_NO_OUTPUT • 0x04: IO_CAP_KEYBOARD_DISPLAY

Output parameters

Table 144. **ACI_GAP_SET_IO_CAPABILITY output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.6 ACI_GAP_SET_AUTHENTICATION_REQUIREMENT

Description

Set the authentication requirements for the device. If the OOB_Enable is set to 0, the following 16 octets of OOB_Data are ignored on reception. This command has to be given only when the device is not in a connected state.

Input parameters

Table 145. **ACI_GAP_SET_AUTHENTICATION_REQUIREMENT input parameters**

Parameter	Size	Description	Possible values
Bonding_Mode	1	Bonding mode. Only if bonding is enabled (0x01), the bonding information is stored in flash	• 0x00: No-bonding mode • 0x01: Bonding mode
MITM_Mode	1	MITM mode	• 0x00: MITM protection not required • 0x01: MITM protection required
SC_Support	1	LE secure connections support	• 0x00: Secure connections pairing not supported • 0x01: Secure connections pairing supported but optional • 0x02: Secure connections pairing supported and mandatory (SC only mode)
KeyPress_Notification_Support	1	Keypress notification support	• 0x00: Keypress notification not supported • 0x01: Keypress notification supported
Min_Encryption_Key_Size	1	Minimum encryption key size to be used during pairing	-
Max_Encryption_Key_Size	1	Maximum encryption key size to be used during pairing	-
Use_Fixed_Pin	1	Use or not fixed pin. If set to 0x00, then during the pairing process the application is not requested for a pin (Fixed_Pin is used). If set to 0x01, then during pairing process if a passkey is required the application is notified	• 0x00: use a fixed pin • 0x01: do not use a fixed pin
Fixed_Pin	4	Fixed pin to be used during pairing if MIMT protection is enabled. Any random value between 0 to 999999	0 ... 999999
Identity_Address_Type	1	Identity address type	• 0x00: Public identity address • 0x01: Random (static) identity address

Output parameters

Table 146. **ACI_GAP_SET_AUTHENTICATION_REQUIREMENT output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.7 ACI_GAP_SET_AUTHORIZATION_REQUIREMENT

Description

Set the authorization requirements of the device. This command has to be given when connected to a device if authorization is required to access services which require authorization.

Input parameters

Table 147. ACI_GAP_SET_AUTHORIZATION_REQUIREMENT input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Authorization_Enable	1	Enable the authorization in the device and when a remote device tries to read/write a characteristic with authorization requirements, the stack sends back an error response with "Insufficient authorization" error code. After pairing is complete a ACI_GAP_AUTHORIZATION_REQ_EVENT is sent to the host.	• 0x00: Authorization not required • 0x01: Authorization required

Output parameters

Table 148. ACI_GAP_SET_AUTHORIZATION_REQUIREMENT output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.8 ACI_GAP_PASS_KEY_RESP

Description

This command is sent by the host in response to ACI_GAP_PASS_KEY_REQ_EVENT event. The command parameter contains the pass key which is used during the pairing process.

Input parameters

Table 149. ACI_GAP_PASS_KEY_RESP input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Pass_Key	4	Pass key that is used during the pairing process. Must be a six-digit decimal number.	0 ... 999999

Output parameters

Table 150. ACI_GAP_PASS_KEY_RESP output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.9 ACI_GAP_AUTHORIZATION_RESP

Description

Authorize a device to access attributes. This command is sent by the host in response to ACI_GAP_AUTHORIZATION_REQ_EVENT event.

Input parameters

Table 151. ACI_GAP_AUTHORIZATION_RESP input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Authorize	1	Authorization response.	• 0x01: Authorize • 0x02: Reject

Output parameters

Table 152. ACI_GAP_AUTHORIZATION_RESP output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.10 ACI_GAP_INIT

Description

Initialize the GAP layer. Register the GAP service with the GATT. All the standard GAP characteristics are added:

- Device Name
- Appearance
- Peripheral Preferred Connection Parameters (peripheral role only)

Note: If the Peripheral Preferred Connection Parameters characteristic is added, its handle is equal to the Appearance characteristic handle plus '2'.

Input parameters

Table 153. ACI_GAP_INIT input parameters

Parameter	Size	Description	Possible values
Role	1	Bitmap of allowed roles.	Bitmask of: • 0x01: Peripheral • 0x02: Broadcaster • 0x04: Central • 0x08: Observer
privacy_enabled	1	Specify if privacy is enabled or not and which one	• 0x00: Privacy disabled • 0x01: Privacy host enabled – 0x02: Privacy controller enabled
device_name_char_len	1	Length of the device name characteristic	-

Output parameters

Table 154. **ACI_GAP_INIT output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Service_Handle	2	Handle of the GAP service	-
Dev_Name_Char_Handle	2	Device name characteristic handle	-
Appearance_Char_Handle	2	Appearance characteristic handle	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.11 ACI_GAP_SET_NON_CONNECTABLE

Description

Put the device into non connectable mode. This mode does not support connection. The privacy setting done in the ACI_GAP_INIT command plays a role in deciding the valid parameters for this command. Advertiser filter policy is internally set to 0x00.

Input parameters

Table 155. **ACI_GAP_SET_NON_CONNECTABLE input parameters**

Parameter	Size	Description	Possible values
Advertising_Event_Type	1	Advertising type.	• 0x02: ADV_SCAN_IND (Scannable undirected advertising) • 0x03: ADV_NONCONN_IND (Non connectable undirected advertising)
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address

Output parameters

Table 156. **ACI_GAP_SET_NON_CONNECTABLE output parameters**

Parameter	Size	Description	Possible values
Status	1	Error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.12 ACI_GAP_SET_UNDIRECTED_CONNECTABLE

Description

Put the device into undirected connectable mode. If privacy is enabled in the device, a resolvable private address is generated and used as the advertiser's address. If not, the address of the type specified in own_addr_type is used for advertising.

Input parameters

Table 157. **ACI_GAP_SET_UNDIRECTED_CONNECTABLE input parameters**

Parameter	Size	Description	Possible values
Advertising_Interval_Min	2	Minimum advertising interval. Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if controller privacy is enabled or if host privacy (i.e. privacy 1.1) is enabled) • 0x03: Non Resolvable private address (it is allowed only if host privacy (i.e. privacy 1.1) is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address
Adv_Filter_Policy	1	Advertising filter policy.	• 0x00: Allow scan request from any, allow connect request from any • 0x03: Allow scan request from white list only, allow connect request from white list only

Output parameters

Table 158. **ACI_GAP_SET_UNDIRECTED_CONNECTABLE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.13 ACI_GAP_SLAVE_SECURITY_REQ

Description

Send a slave security request to the master. This command has to be issued to notify the master of the security requirements of the slave. The master may encrypt the link, initiate the pairing procedure, or reject the request.

Input parameters

Table 159. **ACI_GAP_SLAVE_SECURITY_REQ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 160. **ACI_GAP_SLAVE_SECURITY_REQ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GAP_SLAVE_SECURITY_INITIATED_EVENT

2.4.14 ACI_GAP_UPDATE_ADV_DATA

Description

This command can be used to update the advertising data for a particular AD type. If the AD type specified does not exist, then it is added to the advertising data. If the overall advertising data length is more than 31 octets after the update, then the command is rejected and the old data is retained.

Input parameters

Table 161. ACI_GAP_UPDATE_ADV_DATA input parameters

Parameter	Size	Description	Possible values
AdvDataLen	1	Length of AdvData in octets	-
AdvData	AdvDataLen	Advertising data used by the device while advertising.	-

Output parameters

Table 162. ACI_GAP_UPDATE_ADV_DATA output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.15 ACI_GAP_DELETE_AD_TYPE

Description

This command can be used to delete the specified AD type from the advertisement data if present.

Input parameters

Table 163. ACI_GAP_DELETE_AD_TYPE input parameters

Parameter	Size	Description	Possible values
ADType	1	One of the AD types like in Bluetooth specification	-

Output parameters

Table 164. ACI_GAP_DELETE_AD_TYPE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.16 ACI_GAP_GET_SECURITY_LEVEL

Description

This command can be used to get the current security settings of the device.

Input parameters

Table 165. **ACI_GAP_GET_SECURITY_LEVEL input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 166. **ACI_GAP_GET_SECURITY_LEVEL output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Security_Mode	1	Security mode.	• 0x01: Security Mode 1 • 0x02: Security Mode 2
Security_Level	1	Security level.	• 0x01: Security Level 1 • 0x02: Security Level 2 • 0x03: Security Level 3 • 0x04: Security Level 4

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.17 ACI_GAP_SET_EVENT_MASK

Description

It allows masking events from the GAP. The default configuration is all the events masked.

Input parameters

Table 167. **ACI_GAP_SET_EVENT_MASK input parameters**

Parameter	Size	Description	Possible values
GAP_Evt_Mask	2	GAP event mask. Default: 0xFFFF	Bitmask of: • 0x0000: No events • 0x0001: ACI_GAP_LIMITED_DISCOVERABLE_EVENT • 0x0002: ACI_GAP_PAIRING_COMPLETE_EVENT • 0x0004: ACI_GAP_PASS_KEY_REQ_EVENT • 0x0008: ACI_GAP_AUTHORIZATION_REQ_EVENT • 0x0010: ACI_GAP_SLAVE_SECURITY_INITIATED_EVENT • 0x0020: ACI_GAP_BOND_LOST_EVENT • 0x0080: ACI_GAP_PROC_COMPLETE_EVENT • 0x0100: ACI_L2CAP_CONNECTION_UPDATE_REQ_EVENT • 0x0200: ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT • 0x0400: ACI_L2CAP_PROC_TIMEOUT_EVENT • 0x0800: ACI_GAP_ADDR_NOT_RESOLVED_EVENT

Output parameters

Table 168. **ACI_GAP_SET_EVENT_MASK output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.18 ACI_GAP_CONFIGURE_WHITELIST

Description

Add addresses of bonded devices into the controller's white-list. The command returns an error if it was unable to add the bonded devices into the whitelist.

Input parameters

None

Output parameters

Table 169. ACI_GAP_CONFIGURE_WHITELIST output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.19 ACI_GAP_TERMINATE

Description

Command the controller to terminate the connection. A HCI_DISCONNECTION_COMPLETE_EVENT event is generated when the link is disconnected. It is important to leave an 100 ms blank window before sending any new command (including system hardware reset), since immediately after HCI_DISCONNECTION_COMPLETE_EVENT event, system save important information in non volatile memory.

Input parameters

Table 170. ACI_GAP_TERMINATE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Reason	1	The reason for ending the connection.	• 0x05: Authentication failure • 0x13: Remote user terminated connection • 0x14: Remote device terminated connection due to low resources • 0x15: Remote device terminated connection due to power-off • 0x1A: Unsupported remote feature • 0x3B: Unacceptable connection parameters

Output parameters

Table 171. ACI_GAP_TERMINATE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_DISCONNECTION_COMPLETE_EVENT

2.4.20 ACI_GAP_CLEAR_SECURITY_DB

Description

Clear the security database. All devices in the security database are removed.

Input parameters

None

Output parameters

Table 172. **ACI_GAP_CLEAR_SECURITY_DB output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.21 ACI_GAP_ALLOW_REBOND

Description

Allows the security manager to complete the pairing procedure and re-bond with the master. This command is given by the application when it receives the ACI_GAP_BOND_LOST_EVENT if it wants the re-bonding to happen successfully. If this command is not given on receiving the event, the bonding procedure timeouts.

Input parameters

Table 173. **ACI_GAP_ALLOW_REBOND input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 174. **ACI_GAP_ALLOW_REBOND output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.22 ACI_GAP_START_LIMITED_DISCOVERY_PROC

Description

Start the limited discovery procedure. The controller starts active scanning, only the devices in limited discoverable mode are returned to the upper layers. The procedure is terminated when either the upper layers issue a command to terminate the procedure by issuing the command ACI_GAP_TERMINATE_GAP_PROC with the procedure code set to 0x01, or a there is a timeout (the timeout value is fixed at 10.24 s). When the procedure is terminated for any of the above reasons, ACI_GAP_PROC_COMPLETE_EVENT event is returned with the procedure code set to 0x01. The device found when the procedure is ongoing is returned to the upper layers through the event HCI_LE_ADVERTISING_REPORT_EVENT.

Input parameters

Table 175. **ACI_GAP_START_LIMITED_DISCOVERY_PROC input parameters**

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	This is defined as the time interval from when the Controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window is less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address
Filter_Duplicates	1	Enable/disable duplicate filtering.	• 0x00: Duplicate filtering disabled • 0x01: Duplicate filtering enabled

Output parameters

Table 176. **ACI_GAP_START_LIMITED_DISCOVERY_PROC output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_ADVERTISING_REPORT_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.4.23 ACI_GAP_START_GENERAL_DISCOVERY_PROC

Description

Start the general discovery procedure. The controller is commanded to start active scanning. The procedure is terminated when either the upper layers issue a command to terminate the procedure by issuing the command ACI_GAP_TERMINATE_GAP_PROC with the procedure code set to 0x02 or a timeout happens (the timeout value is fixed at 10.24 s). When the procedure is terminated due to any of the above reasons, ACI_GAP_PROC_COMPLETE_EVENT event is returned with the procedure code set to 0x02.

The device found when the procedure is ongoing is returned to HCI_LE_ADVERTISING_REPORT_EVENT.

Input parameters

Table 177. **ACI_GAP_START_GENERAL_DISCOVERY_PROC input parameters**

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	This is defined as the time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window is less than or equal to LE_Scan_Interval.	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)

Parameter	Size	Description	Possible values
		Time = N * 0.625 ms	
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address
Filter_Duplicates	1	Enable/disable duplicate filtering.	• 0x00: Duplicate filtering disabled • 0x01: Duplicate filtering enabled

Output parameters

Table 178. ACI_GAP_START_GENERAL_DISCOVERY_PROC output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_ADVERTISING_REPORT_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.4.24 ACI_GAP_START_NAME_DISCOVERY_PROC

Description

Start the name discovery procedure. A LE_Create_Connection call is made to the controller by GAP with the initiator filter policy set to "ignore whitelist and process connectable advertising packets only for the specified device". Once a connection is established, GATT procedure is started to read the device name characteristic. When the read is completed (successfully or unsuccessfully), a ACI_GAP_PROC_COMPLETE_EVENT event is given to the upper layer. The event also contains the name of the device if the device name was read successfully.

Input parameters

Table 179. ACI_GAP_START_NAME_DISCOVERY_PROC input parameters

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	This is defined as the time interval from when the Controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window is less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
Peer_Address_Type	1	Address type.	• 0x00: Public device address • 0x01: Random device address
Peer_Address	6	Public device address or random device address of the device to be connected.	-

Parameter	Size	Description	Possible values
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address
Conn_Interval_Min	2	Minimum value for the connection event interval. This is less than or equal to Conn_Interval_Max. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Interval_Max	2	Maximum value for the connection event interval. This is greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE Link. It is a multiple of 10 ms and larger than (1 + connSlaveLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)
Minimum_CE_Length	2	Information parameter about the minimum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Maximum_CE_Length	2	Information parameter about the maximum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)

Output parameters

Table 180. ACI_GAP_START_NAME_DISCOVERY_PROC output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.4.25 ACI_GAP_START_AUTO_CONNECTION_ESTABLISH_PROC

Description

Start the auto connection establishment procedure. The devices specified are added to the white list of the controller and a LE_Create_Connection call is made to the controller by GAP with the initiator filter policy set to "use whitelist to determine which advertiser to connect to". When a command is issued to terminate the procedure by upper layer, a LE_Create_Connection_Cancel call is made to the controller by GAP. The procedure is terminated when either a connection is successfully established with one of the specified devices in the white list or the procedure is explicitly terminated by issuing the command ACI_GAP_TERMINATE_GAP_PROC with the procedure code set to 0x08. A ACI_GAP_PROC_COMPLETE_EVENT event is returned with the procedure code set to 0x08. If controller privacy is enabled and the peer device (advertiser) is in the resolving list then the link layer generates a RPA, if it is not then the RPA/NRPA generated by the host is used.

Input parameters

Table 181. **ACI_GAP_START_AUTO_CONNECTION_ESTABLISH_PROC input parameters**

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	This is defined as the time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window is less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address
Conn_Interval_Min	2	Minimum value for the connection event interval. This is less than or equal to Conn_Interval_Max. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Interval_Max	2	Maximum value for the connection event interval. This is greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE Link. It is a multiple of 10 ms and larger than (1 + connSlaveLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)
Minimum_CE_Length	2	Information parameter about the minimum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Maximum_CE_Length	2	Information parameter about the maximum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Num_of_Whitelist_Entries	1	Number of devices that have to be added to the white list.	-
Peer_Address_Type[i]	1	Address type.	• 0x00: Public device address • 0x01: Random device address
Peer_Address[i]	6	Public device address or random device address of the device to be added to the white list.	-

Output parameters

Table 182. **ACI_GAP_START_AUTO_CONNECTION_ESTABLISH_PROC output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT

- ACI_GAP_PROC_COMPLETE_EVENT

2.4.26 ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC

Description

Start a general connection establishment procedure. The host enables scanning in the controller with the scanner filter policy set to "accept all advertising packets" and from the scanning results, all the devices are sent to the upper layer using the event LE_Advertising_Report. The upper layer then has to select the device to which it wants to connect by issuing the command ACI_GAP_CREATE_CONNECTION. If privacy is enabled, then either a private resolvable address or a non resolvable address, based on the address type specified in the command is set as the scanner address, but the gap create connection always uses a private resolvable address if the general connection establishment procedure is active. The procedure is terminated when a connection is established or the upper layer terminates it by issuing the command ACI_GAP_TERMINATE_GAP_PROC with the procedure code set to 0x10. On procedure completion a ACI_GAP_PROC_COMPLETE_EVENT event is generated with the procedure code set to 0x10. If controller privacy is enabled and the peer device (advertiser) is in the resolving list then the link layer generates a RPA, if it is not the RPA/NRPA generated by the host is used.

Input parameters

Table 183. ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC input parameters

Parameter	Size	Description	Possible values
LE_Scan_Type	1	Passive or active scanning. With active scanning SCAN_REQ packets are sent.	• 0x00: Passive scanning • 0x01: Active scanning
LE_Scan_Interval	2	This is defined as the time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window is less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address
Scanning_Filter_Policy	1	Scanning filter policy: • 0x00 Accept all advertisement packets.Directed advertising packets which are not addressed for this device is ignored. • 0x01 Ignore advertisement packets from devices not in the white list Only. Directed advertising packets which are not addressed for this device is ignored. • 0x02 Accept all undirected advertisement packets (it is allowed only if controller privacy or host privacy is enabled). Directed advertisement packets where initiator address is a RPA and directed advertisement packets addressed to this device is accepted. • 0x03 Accept all undirected advertisement packets from devices that are in the white list. Directed advertisement packets where initiator address is RPA and directed advertisement packets addressed to this device is accepted.	• 0x00: Accept all • 0x01: Ignore devices not in the White List • 0x02: Accept all (use resolving list) • 0x03: Ignore devices not in the white list (use resolving list)

Parameter	Size	Description	Possible values
		• NOTE: if controller privacy is enabled Scanning_Filter_Policy can only assume values 0x00 or 0x02; if host privacy is enabled Scanning_Filter_Policy can only assume value 0x00.	
Filter_Duplicates	1	Enable/disable duplicate filtering.	• 0x00: Duplicate filtering disabled • 0x01: Duplicate filtering enabled

Output parameters

Table 184. ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.4.27 ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC

Description

Start a selective connection establishment procedure. The GAP adds the specified device addresses into white list and enables scanning in the controller with the scanner filter policy set to "accept packets only from devices in white list". All the devices found are sent to the upper layer by the event HCI_LE_ADVERTISING_REPORT_EVENT. The upper layer then has to select one of the devices to which it wants to connect by issuing the command ACI_GAP_CREATE_CONNECTION. On completion of the procedure a ACI_GAP_PROC_COMPLETE_EVENT event is generated with the procedure code set to 0x20. The procedure is terminated when a connection is established or the upper layer terminates the procedure by issuing the command ACI_GAP_TERMINATE_GAP_PROC with the procedure code set to 0x20. If controller privacy is enabled and the peer device (advertiser) is in the resolving list then the link layer generates a RPA, if it is not then the RPA/NRPA generated by the host is used.

Input parameters

Table 185. ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC input parameters

Parameter	Size	Description	Possible values
LE_Scan_Type	1	Passive or active scanning. With active scanning SCAN_REQ packets are sent.	• 0x00: Passive Scanning • 0x01: Active scanning
LE_Scan_Interval	2	This is defined as the time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window is less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address

Parameter	Size	Description	Possible values
		• 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x03: Non resolvable private address
Scanning_Filter_Policy	1	Scanning filter policy: • 0x00 Accept all advertisement packets. Directed advertising packets which are not addressed for this device is ignored. • 0x01 Ignore advertisement packets from devices not in the white list Only. Directed advertising packets which are not addressed for this device is ignored. • 0x02 Accept all undirected advertisement packets (it is allowed only if controller privacy or host privacy is enabled). Directed advertisement packets where initiator address is a RPA and directed advertisement packets addressed to this device is accepted. • 0x03 Accept all undirected advertisement packets from devices that are in the white list. Directed advertisement packets where initiator address is RPA and directed advertisement packets addressed to this device is accepted. *Note: if controller privacy is enabled Scanning_Filter_Policy can only assume values 0x01 or 0x03; if host privacy is enabled Scanning_Filter_Policy can only assume value 0x01.*	• 0x00: Accept all • 0x01: Ignore devices not in the white list • 0x02: Accept all (use resolving list) • 0x03: Ignore devices not in the white list (use resolving list)
Filter_Duplicates	1	Enable/disable duplicate filtering.	• 0x00: Duplicate filtering disabled • 0x01: Duplicate filtering enabled
Num_of_Whitelist_Entries	1	Number of devices that have to be added to the white list.	-
Peer_Address_Type[i]	1	Address type.	• 0x00: Public device address • 0x01: Random device address
Peer_Address[i]	6	Public device address or random device address of the device to be added to the white list.	-

Output parameters

Table 186. ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.4.28 ACI_GAP_CREATE_CONNECTION

Description

Start the direct connection establishment procedure. A LE_Create_Connection call is made to the controller by GAP with the initiator filter policy set to "ignore white list and process connectable advertising packets only for the specified device".

The procedure can be terminated explicitly by the upper layer by issuing the command ACI_GAP_TERMINATE_GAP_PROC. When a command is issued to terminate the procedure by upper layer, a HCI_LE_CREATE_CONNECTION_CANCEL call is made to the controller by GAP. On termination of the procedure, a HCI_LE_CONNECTION_COMPLETE_EVENT event is returned. The procedure can be explicitly terminated by the upper layer by issuing the command ACI_GAP_TERMINATE_GAP_PROC with the procedure_code set to 0x40. If controller privacy is enabled and the peer device (advertiser) is in the resolving list then the link layer generates a RPA, if it is not then the RPA/NRPA generated by the host is used.

Input parameters

Table 187. ACI_GAP_CREATE_CONNECTION input parameters

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	This is defined as the time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window shall be less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
Peer_Address_Type	1	The address type of the peer device.	• 0x00: Public device address • 0x01: Random device address
Peer_Address	6	Public device address or random device address of the device to be connected.	-
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address
Conn_Interval_Min	2	Minimum value for the connection event interval. This shall be less than or equal to Conn_Interval_Max. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Interval_Max	2	Maximum value for the connection event interval. This shall be greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE link. It shall be a multiple of 10 ms and larger than (1 + connSlaveLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)
Minimum_CE_Length	2	Information parameter about the minimum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Maximum_CE_Length	2	Information parameter about the maximum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)

Output parameters

Table 188. **ACI_GAP_CREATE_CONNECTION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_CONNECTION_COMPLETE_EVENT

2.4.29 ACI_GAP_TERMINATE_GAP_PROC

Description

Terminate the specified GATT procedure. An ACI_GAP_PROC_COMPLETE_EVENT event is returned with the procedure code set to the corresponding procedure.

Input parameters

Table 189. **ACI_GAP_TERMINATE_GAP_PROC input parameters**

Parameter	Size	Description	Possible values
Procedure_Code	1	GAP procedure bitmap.	• 0x00: No events • 0x01:GAP_ LIMITED_DISCOVERY_PROC • 0x02:GAP_GENERAL_DISCOVERY_PROC • 0x04: GAP_NAME_DISCOVERY_PROC • 0x08: GAP_AUTO_CONNECTION_ESTABLISHMENT_PROC • 0x10: GAP_GENERAL_CONNECTION_ESTABLISHMENT_PROC • 0x20: GAP_SELECTIVE_CONNECTION_ESTABLISHMENT_PROC • 0x40: GAP_DIRECT_CONNECTION_ESTABLISHMENT_PROC • 0x80: GAP_OBSERVATION_PROC

Output parameters

Table 190. **ACI_GAP_TERMINATE_GAP_PROC output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.4.30 ACI_GAP_START_CONNECTION_UPDATE

Description

Start the connection update procedure (only when role is Master). A HCI_LE_CONNECTION_UPDATE is called. On completion of the procedure, an HCI_LE_CONNECTION_UPDATE_COMPLETE_EVENT event is returned to the upper layer.

Input parameters

Table 191. **ACI_GAP_START_CONNECTION_UPDATE input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Conn_Interval_Min	2	Minimum value for the connection event interval. This is less than or equal to Conn_Interval_Max.	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)

Parameter	Size	Description	Possible values
		Time = N * 1.25 ms	
Conn_Interval_Max	2	Maximum value for the connection event interval. This is greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE link. It is a multiple of 10 ms and larger than (1 + connSlaveLatency) * connInterval * 2. Time = N * 10 msec.	0x000A (100 ms) ... 0x0C80 (32000 ms)
Minimum_CE_Length	2	Information parameter about the minimum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Maximum_CE_Length	2	Information parameter about the maximum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)

Output parameters

Table 192. ACI_GAP_START_CONNECTION_UPDATE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_CONNECTION_UPDATE_COMPLETE_EVENT

2.4.31 ACI_GAP_SEND_PAIRING_REQ

Description

Send the SM pairing request to start a pairing process. The authentication requirements and IO capabilities must be set before issuing this command using the ACI_GAP_SET_IO_CAPABILITY and ACI_GAP_SET_AUTHENTICATION_REQUIREMENT commands. A ACI_GAP_PAIRING_COMPLETE_EVENT event is returned after the pairing process is completed.

Input parameters

Table 193. ACI_GAP_SEND_PAIRING_REQ input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Force_Rebond	1	If 1, pairing request is sent even if the device was previously bonded, otherwise pairing request is not sent.	• 0x00: NO • 0x01: YES

Output parameters

Table 194. ACI_GAP_SEND_PAIRING_REQ output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT

- ACI_GAP_PAIRING_COMPLETE_EVENT

2.4.32 ACI_GAP_RESOLVE_PRIVATE_ADDR

Description

This command tries to resolve the address provided with the IRKs present in its database. If the address is resolved successfully with any one of the IRKs present in the database, it returns success and also the corresponding public/static random address stored with the IRK in the database.

Input parameters

Table 195. ACI_GAP_RESOLVE_PRIVATE_ADDR input parameters

Parameter	Size	Description	Possible values
Address	6	Address to be resolved	-

Output parameters

Table 196. ACI_GAP_RESOLVE_PRIVATE_ADDR output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Actual_Address	6	The public or static random address of the peer device, distributed during pairing phase.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.33 ACI_GAP_SET_BROADCAST_MODE

Description

This command puts the device into broadcast mode. A privacy enabled device uses either a resolvable private address or a non-resolvable private address as specified in the Own_Addr_Type parameter of the command.

Input parameters

Table 197. ACI_GAP_SET_BROADCAST_MODE input parameters

Parameter	Size	Description	Possible values
Advertising_Interval_Min	2	Minimum advertising interval. Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 ms	0x0020 (20.000 ms) ... 0x4000 (10240.000 ms)
Advertising_Type	1	Non connectable advertising type	• 0x02: ADV_SCAN_IND (Scannable undirected advertising) • 0x03: ADV_NONCONN_IND (Non connectable undirected advertising)
Own_Address_Type	1	If privacy is disabled, then the address can be public or static random. If privacy is enabled, then the address can be a resolvable private address or a non-resolvable private address.	• 0x00: Public address • 0x01: Static random address • 0x02: Resolvable private address • 0x03: Non-resolvable private address
Adv_Data_Length	1	Length of the advertising data in the advertising packet.	-

Parameter	Size	Description	Possible values
Adv_Data	Adv_Data_Length	Advertising data used by the device while advertising.	-
Num_of_Whitelist_Entries	1	Number of devices that have to be added to the white list.	-
Peer_Address_Type[i]	1	Address type.	• 0x00: Public device address • 0x01: Random device address
Peer_Address[i]	6	Public device address or random device address of the device to be added to the white list.	-

Output parameters

Table 198. ACI_GAP_SET_BROADCAST_MODE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.34 ACI_GAP_START_OBSERVATION_PROC

Description

Starts an observation procedure, when the device is in observer role. The host enables scanning in the controller. The advertising reports are sent to the upper layer using standard LE advertising report event. If controller privacy is enabled and the peer device (advertiser) is in the resolving list then the link layer generates a RPA, if it is not then the RPA/NRPA generated by the host is used.

Input parameters

Table 199. ACI_GAP_START_OBSERVATION_PROC input parameters

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	This is defined as the time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window is less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	0x0004 (2.500 ms) ... 0x4000 (10240.000 ms)
LE_Scan_Type	1	Passive or active scanning. With active scanning SCAN_REQ packets are sent.	• 0x00: Passive scanning • 0x01: Active scanning
Own_Address_Type	1	Own address type: • 0x00: Public device address (it is allowed only if privacy is disabled) • 0x01: Random device address (it is allowed only if privacy is disabled) • 0x02: Resolvable private address (it is allowed only if privacy is enabled) • 0x03: Non resolvable private address (it is allowed only if privacy is enabled)	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address
Filter_Duplicates	1	Enable/disable duplicate filtering.	• 0x00: Duplicate filtering disabled • 0x01: Duplicate filtering enabled
Scanning_Filter_Policy	1	Scanning filter policy: • 0x00 Accept all advertisement packets (it is allowed only if controller privacy is enabled). Directed advertising packets which are not addressed for this device is ignored. • 0x01 Ignore advertisement packets from devices not in the white list Only. Directed advertising packets which are not addressed for this device is ignored. • 0x02 Accept all undirected advertisement packets (it is allowed only if controller privacy or host privacy is enabled). Directed advertisement packets where initiator address is a RPA and directed advertisement packets addressed to this device is accepted. • 0x03 Accept all undirected advertisement packets from devices that are in the white list. Directed advertisement packets where initiator address is RPA and directed advertisement packets addressed to this device is accepted. *Note: If host privacy is enabled Scanning_Filter_Policy can only take values 0x00 or 0x01.*	• 0x00: Accept all • 0x01: Ignore devices not in the white list • 0x02: Accept all (use resolving list) • 0x03: Ignore devices not in the white list (use resolving list)

Output parameters

Table 200. ACI_GAP_START_OBSERVATION_PROC output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_ADVERTISING_REPORT_EVENT

2.4.35 ACI_GAP_GET_BONDED_DEVICES

Description

This command gets the list of the devices which are bonded. It returns the number of addresses and the corresponding address types and values.

Input parameters

None

Output parameters

Table 201. **ACI_GAP_GET_BONDED_DEVICES output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Num_of_Addresses	1	The number of bonded devices	-
Address_Type[i]	1	Address type.	• 0x00: Public device address • 0x01: Random device address
Address[i]	6	Public device address or random device address of the device to be added to the white list.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.36 ACI_GAP_IS_DEVICE_BONDED

Description

The command finds whether the device, whose address is specified in the command, is bonded. If the device is using a resolvable private address and it has been bonded, then the command returns BLE_STATUS_SUCCESS.

Input parameters

Table 202. **ACI_GAP_IS_DEVICE_BONDED input parameters**

Parameter	Size	Description	Possible values
Peer_Address_Type	1	Address type.	• 0x00: Public device address • 0x01: Random device address
Peer_Address	6	Address used by the peer device while advertising	-

Output parameters

Table 203. **ACI_GAP_IS_DEVICE_BONDED output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.37 ACI_GAP_NUMERIC_COMPARISON_VALUE_CONFIRM_YESNO

Description

This command allows the user to validate/confirm or not the Numeric Comparison value shown through the ACI_GAP_Numeric_Comparison_Value_Event.

Input parameters

Table 204. **ACI_GAP_NUMERIC_COMPARISON_VALUE_CONFIRM_YESNO input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Confirm_Yes_No	1	0 : The Numeric values showed on both local and peer device are different 1 : The Numeric values showed on both local and peer device are equal!	• 0x00: No • 0x01: Yes

Output parameters

Table 205. **ACI_GAP_NUMERIC_COMPARISON_VALUE_CONFIRM_YESNO output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.38 ACI_GAP_PASSKEY_INPUT

Description

This command permits to signal to the stack the input type detected during passkey input.

Input parameters

Table 206. **ACI_GAP_PASSKEY_INPUT input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Input_Type	1	Passkey input type detected	• 0x00: Passkey entry started • 0x01: Passkey digit entered • 0x02: Passkey digit erased • 0x03: Passkey cleared • 0x04: Passkey entry completed

Output parameters

Table 207. **ACI_GAP_PASSKEY_INPUT output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.39 ACI_GAP_GET_OOB_DATA

Description

This command is sent by the User to get (i.e. to extract from the stack) the OOB data generated by the stack itself.

Input parameters

Table 208. **ACI_GAP_GET_OOB_DATA input parameters**

Parameter	Size	Description	Possible values
OOB_Data_Type	1	OOB data type	• 0x00: TK • 0x01: Random • 0x02: Confirm

Output parameters

Table 209. **ACI_GAP_GET_OOB_DATA output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Address_Type	1	Identity address type.	• 0x00: Public identity address • 0x01: Random (static) identity address
Address	6	Public or random (static) address of this device	
OOB_Data_Type	1	OOB data type	• 0x00: TK • 0x01: Random • 0x02: Confirm
OOB_Data_Len	1	Length of OOB data	-
OOB_Data	16	Local pairing data intended to the remote device to be sent via OOB.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.40 ACI_GAP_SET_OOB_DATA

Description

This command is sent (by the user) to input the OOB data arrived via OOB communication.

Input parameters

Table 210. **ACI_GAP_SET_OOB_DATA input parameters**

Parameter	Size	Description	Possible values
Device_Type	1	OOB Device type	• 0x00: Local device • 0x01: Remote device
Address_Type	1	Identity address type.	• 0x00: Public identity address • 0x01: Random (static) identity address
Address	6	Public or random (static) address of the peer device	
OOB_Data_Type	1	OOB data type	• 0x00: TK • 0x01: Random • 0x02: Confirm
OOB_Data_Len	1	Length of OOB data	-
OOB_Data	16	Pairing data received through OOB from remote device	-

Output parameters

Table 211. **ACI_GAP_SET_OOB_DATA output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.41 ACI_GAP_ADD_DEVICES_TO_RESOLVING_LIST

Description

This command is used to add one device to the list of address translations used to resolve resolvable private addresses in the controller.

Input parameters

Table 212. **ACI_GAP_ADD_DEVICES_TO_RESOLVING_LIST input parameters**

Parameter	Size	Description	Possible values
Num_of_Resolving_list_Entries	1	Number of devices that have to be added to the resolving list.	-
Peer_Identity_Address_Type[i]	1	Identity address type.	• 0x00: Public identity address • 0x01: Random (static) identity address
Peer_Identity_Address[i]	6	Public or random (static) identity address of the peer device	-
Clear_Resolving_List	1	Clear the resolving list	-

Output parameters

Table 213. **ACI_GAP_ADD_DEVICES_TO_RESOLVING_LIST output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.42 ACI_GAP_REMOVE_BONDED_DEVICE

Description

This command is used to remove a specified device from bonding table.

Input parameters

Table 214. **ACI_GAP_REMOVE_BONDED_DEVICE input parameters**

Parameter	Size	Description	Possible values
Peer_Identity_Address_Type	1	Identity address type.	• 0x00: Public identity address • 0x01: Random (static) identity address
Peer_Identity_Address	6	Public or random (static) identity address of the peer device	-

Output parameters

Table 215. **ACI_GAP_REMOVE_BONDED_DEVICE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5 GATT/ATT commands

In Table 216 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / SO, respectively, for Link layer only / Beacon only / Slave only.

Table 216. GATT/ATT commands list

Command	OpCode	LO	SO	BO
ACI_GATT_INIT	0xFD01	-	Y	-
ACI_GATT_ADD_SERVICE	0xFD02	-	Y	-
ACI_GATT_INCLUDE_SERVICE	0xFD03	-	Y	-
ACI_GATT_ADD_CHAR	0xFD04	-	Y	-
ACI_GATT_ADD_CHAR_DESC	0xFD05	-	Y	-
ACI_GATT_UPDATE_CHAR_VALUE	0xFD06	-	Y	-
ACI_GATT_DEL_CHAR	0xFD07	-	Y	-
ACI_GATT_DEL_SERVICE	0xFD08	-	Y	-
ACI_GATT_DEL_INCLUDE_SERVICE	0xFD09	-	Y	-
ACI_GATT_SET_EVENT_MASK	0xFD0A	-	Y	-
ACI_GATT_EXCHANGE_CONFIG	0xFD0B	-	-	-
ACI_ATT_FIND_INFO_REQ	0xFD0C	-	-	-
ACI_ATT_FIND_BY_TYPE_VALUE_REQ	0xFD0D	-	-	-
ACI_ATT_READ_BY_TYPE_REQ	0xFD0E	-	-	-
ACI_ATT_READ_BY_GROUP_TYPE_REQ	0xFD0F	-	-	-
ACI_ATT_PREPARE_WRITE_REQ	0xFD10	-	-	-
ACI_ATT_EXECUTE_WRITE_REQ	0xFD11	-	-	-
ACI_GATT_DISC_ALL_PRIMARY_SERVICES	0xFD12	-	-	-
ACI_GATT_DISC_PRIMARY_SERVICE_BY_UUID	0xFD13	-	-	-
ACI_GATT_FIND_INCLUDED_SERVICES	0xFD14	-	-	-
ACI_GATT_DISC_ALL_CHAR_OF_SERVICE	0xFD15	-	-	-
ACI_GATT_DISC_CHAR_BY_UUID	0xFD16	-	-	-
ACI_GATT_DISC_ALL_CHAR_DESC	0xFD17	-	-	-
ACI_GATT_READ_CHAR_VALUE	0xFD18	-	-	-
ACI_GATT_READ_USING_CHAR_UUID	0xFD19	-	-	-
ACI_GATT_READ_LONG_CHAR_VALUE	0xFD1A	-	-	-
ACI_GATT_READ_MULTIPLE_CHAR_VALUE	0xFD1B	-	-	-
ACI_GATT_WRITE_CHAR_VALUE	0xFD1C	-	-	-
ACI_GATT_WRITE_LONG_CHAR_VALUE	0xFD1D	-	-	-
ACI_GATT_WRITE_CHAR_RELIABLE	0xFD1E	-	-	-
ACI_GATT_WRITE_LONG_CHAR_DESC	0xFD1F	-	-	-
ACI_GATT_READ_LONG_CHAR_DESC	0xFD20	-	-	-
ACI_GATT_WRITE_CHAR_DESC	0xFD21	-	-	-
ACI_GATT_READ_CHAR_DESC	0xFD22	-	-	-
ACI_GATT_WRITE_WITHOUT_RESP	0xFD23	-	-	-
ACI_GATT_SIGNED_WRITE_WITHOUT_RESP	0xFD24	-	-	-

Command	OpCode	LO	SO	BO
ACI_GATT_CONFIRM_INDICATION	0xFD25	-	-	-
ACI_GATT_WRITE_RESP	0xFD26	-	Y	-
ACI_GATT_ALLOW_READ	0xFD27	-	Y	-
ACI_GATT_SET_SECURITY_PERMISSION	0xFD28	-	Y	-
ACI_GATT_SET_DESC_VALUE	0xFD29	-	Y	-
ACI_GATT_READ_HANDLE_VALUE	0xFD2A	-	Y	-
ACI_GATT_UPDATE_CHAR_VALUE_EXT	0xFD2C	-	Y	-
ACI_GATT_DENY_READ	0xFD2D	-	Y	-
ACI_GATT_SET_ACCESS_PERMISSION	0xFD2E	-	Y	-

2.5.1 ACI_GATT_INIT

Description

Initialize the GATT layer for server and client roles. It adds also the GATT service with service changed characteristic. Until this command is issued the GATT channel not processes any commands even if the connection is opened. This command has to be given before using any of the GAP features.

Input parameters

None

Output parameters

Table 217. ACI_GATT_INITutput parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.2 ACI_GATT_ADD_SERVICE

Description

Add a service to GATT Server. When a service is created in the server, the host needs to reserve the handle ranges for this service using Max_Attribute_Records parameter. This parameter specifies the maximum number of attribute records that can be added to this service (including the service attribute, include attribute, characteristic attribute, characteristic value attribute and characteristic descriptor attribute). Handle of the created service is returned in command complete event. Service declaration is taken from the service pool. The attributes for characteristics and descriptors are allocated from the attribute pool.

Input parameters

Table 218. ACI_GATT_ADD_SERVICE input parameters

Parameter	Size	Description	Possible values
Service_UUID_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
Service_UUID	2 or 16	16-bit UUID or 128-bit UUID	-
Service_Type	1	Service type	• 0x01: Primary service • 0x02: Secondary service
Max_Attribute_Records	1	Maximum number of attribute records that can be added to this service	-

Output parameters

Table 219. **ACI_GATT_ADD_SERVICE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Service_Handle	2	Handle of the Service. When this service is added, a handle is allocated by the server for this service. Server also allocates a range of handles for this service from serviceHandle to <serviceHandle + max_attr_records - 1>	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.3 ACI_GATT_INCLUDE_SERVICE

Description

Include a service given by Include_Start_Handle and Include_End_Handle to another service given by Service_Handle. Attribute server creates an "include" definition attribute and return the handle of this attribute in Included_handle.

Input parameters

Table 220. **ACI_GATT_INCLUDE_SERVICE input parameters**

Parameter	Size	Description	Possible values
Service_Handle	2	Handle of the service to which another service has to be included.	-
Include_Start_Handle	2	Start handle of the service which has to be included in service	-
Include_End_Handle	2	End handle of the service which has to be included in service	-
Include_UUID_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
Include_UUID	2 or 16	16-bit UUID or 128-bit UUID	-

Output parameters

Table 221. **ACI_GATT_INCLUDE_SERVICEutput parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Include_Handle	2	Handle of the include declaration	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.4 ACI_GATT_ADD_CHAR

Description

Add a characteristic to a service.

Input parameters

Table 222. **ACI_GATT_ADD_CHAR input parameters**

Parameter	Size	Description	Possible values
Service_Handle	2	Handle of the Service to which the characteristic is added	-
Char_UUID_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
Char_UUID	2 or 16	16-bit UUID or 128-bit UUID	-
Char_Value_Length	2	Maximum length of the characteristic value.	-
Char_Properties	1	Characteristic properties	Bitmask of: • 0x00: CHAR_PROP_NONE • 0x01: CHAR_PROP_BROADCAST (Broadcast) • 0x02: CHAR_PROP_READ (Read) • 0x04: CHAR_PROP_WRITE_WITHOUT_RESP (Write w/o resp) • 0x08: CHAR_PROP_WRITE (Write) • 0x10: CHAR_PROP_NOTIFY (Notify) • 0x20: CHAR_PROP_INDICATE (Indicate) • 0x40: CHAR_PROP_SIGNED_WRITE (Authenticated Signed Writes) • 0x80: CHAR_PROP_EXT (Extended Properties)
Security_Permissions	1	Security permission flags.	Bitmask of: • 0x00: None • 0x01: AUTHEN_READ (Need authentication to read) • 0x02: AUTHOR_READ (Need authorization to read) • 0x04: ENCRY_READ (Need encryption to read) • 0x08: AUTHEN_WRITE (need authentication to write) • 0x10: AUTHOR_WRITE (need authorization to write) • 0x20: ENCRY_WRITE (need encryption to write)
GATT_Evt_Mask	1	GATT event mask.	Bitmask of: • 0x00: GATT_DONT_NOTIFY_EVENTS • 0x01: GATT_NOTIFY_ATTRIBUTE_WRITE • 0x02: GATT_NOTIFY_WRITE_REQ_AND_WAIT_FOR_APPL_RESP • 0x04: GATT_NOTIFY_READ_REQ_AND_WAIT_FOR_APPL_RESP
Enc_Key_Size	1	Minimum encryption key size required to read the characteristic.	0x07 ... 0x10
Is_Variable	1	Specify if the characteristic value has a fixed length or a variable length.	• 0x00: Fixed length • 0x01: Variable length

Output parameters

Table 223. **ACI_GATT_ADD_CHAR output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Char_Handle	2	Handle of the Characteristic that has been added. It is the handle of the characteristic declaration. The attribute that holds the characteristic value is allocated at the next handle, followed by the client characteristic configuration descriptor if the characteristic has CHAR_PROP_NOTIFY or CHAR_PROP_INDICATE properties.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.5 ACI_GATT_ADD_CHAR_DESC

Description

Add a characteristic descriptor to a service.

Input parameters

Table 224. **ACI_GATT_ADD_CHAR_DESC input parameters**

Parameter	Size	Description	Possible values
Service_Handle	2	Handle of service to which the characteristic belongs	-
Char_Handle	2	Handle of the characteristic to which description has to be added	-
Char_Desc_Uuid_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
Char_Desc_Uuid	2 or 16	16-bit UUID or 128-bit UUID	-
Char_Desc_Value_ Max_Len	1	The maximum length of the descriptor value	-
Char_Desc_Value_Length	1	Current Length of the characteristic descriptor value	-
Char_Desc_Value	Char_Desc_ Value_Length	Value of the characteristic description	-
Security_Permissions	1	Security permission flags.	Bitmask of: • 0x00: None • 0x01: AUTHEN_READ (Need authentication to read)

Parameter	Size	Description	Possible values
			• 0x02: AUTHOR_READ (Need authorization to read) • 0x04: ENCRY_READ (Need encryption to read) • 0x08: AUTHEN_WRITE (need authentication to write) • 0x10: AUTHOR_WRITE (need authorization to write) • 0x20: ENCRY_WRITE (need encryption to write)
Access_Permissions	1	Access permission	Bitmask of: • 0x00: None • 0x01: READ • 0x02: WRITE • 0x04: WRITE_WO_RESP • 0x08: SIGNED_WRITE
GATT_Evt_Mask	1	GATT event mask.	Bitmask of: • 0x00: GATT_DONT_NOTIFY_EVENTS • 0x01: GATT_NOTIFY_ATTRIBUTE_WRITE • 0x02: GATT_NOTIFY_WRITE_REQ_AND_WAIT_FOR_APPL_RESP • 0x04: GATT_NOTIFY_READ_REQ_AND_WAIT_FOR_APPL_RESP
Enc_Key_Size	1	Minimum encryption key size required to read the characteristic.	0x07 ... 0x10
Is_Variable	1	Specify if the characteristic value has a fixed length or a variable length.	• 0x00: Fixed length • 0x01: Variable length

Output parameters

Table 225. ACI_GATT_ADD_CHAR_DESC output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Char_Desc_Handle	2	Handle of the characteristic descriptor	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.6 ACI_GATT_UPDATE_CHAR_VALUE

Description

Update a characteristic value in a service. If notifications (or indications) are enabled on that characteristic, a notification (or indication) is sent to the client after sending this command to the STM32WB. The command is queued into the STM32WB command queue. If the buffer is full, because previous commands could not be still processed, the function returns BLE_STATUS_INSUFFICIENT_RESOURCES. This happens if notifications (or indications) are enabled and the application calls ACI_GATT_UPDATE_CHAR_VALUE at an higher rate than what is allowed by the link. Throughput on BLE link depends on connection interval and connection length parameters (decided by the master, see aci_l2cap_connection_parameter_update_request() for more info on how to suggest new connection parameters from a slave). If the application does not want to lose notifications because STM32WB buffer becomes full, it has to retry again till the function returns BLE_STATUS_SUCCESS or any other error code.

Input parameters

Table 226. ACI_GATT_UPDATE_CHAR_VALUE input parameters

Parameter	Size	Description	Possible values
Service_Handle	2	Handle of service to which the characteristic belongs	-
Char_Handle	2	Handle of the characteristic declaration	-
Val_Offset	1	The offset from which the attribute value has to be updated. If this is set to 0 and the attribute value is of variable length, then the length of the attribute is set to the Char_Value_Length. If the Val_Offset is set to a value greater than 0, then the length of the attribute is set to the maximum length as specified for the attribute while adding the characteristic.	-
Char_Value_Length	1	Length of the characteristic value in octets	-
Char_Value	Char_Value_Length	Characteristic value	-

Output parameters

Table 227. ACI_GATT_UPDATE_CHAR_VALUE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.7 ACI_GATT_DEL_CHAR

Description

Delete the specified characteristic from the service.

Input parameters

Table 228. ACI_GATT_DEL_CHAR input parameters

Parameter	Size	Description	Possible values
Serv_Handle	2	Handle of service to which the characteristic belongs	-
Char_Handle	2	Handle of the characteristic which has to be deleted	-

Output parameters

Table 229. **ACI_GATT_DEL_CHAR output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.8 ACI_GATT_DEL_SERVICE

Description

Delete the specified service from the GATT server database.

Input parameters

Table 230. **ACI_GATT_DEL_SERVICE input parameters**

Parameter	Size	Description	Possible values
Serv_Handle	2	Handle of the service to be deleted	-

Output parameters

Table 231. **ACI_GATT_DEL_SERVICE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.9 ACI_GATT_DEL_INCLUDE_SERVICE

Description

Delete the include definition from the service.

Input parameters

Table 232. **ACI_GATT_DEL_INCLUDE_SERVICE input parameters**

Parameter	Size	Description	Possible values
Serv_Handle	2	Handle of the service to which the include service belongs	-
Include_Handle	2	Handle of the included service which has to be deleted	-

Output parameters

Table 233. **ACI_GATT_DEL_INCLUDE_SERVICE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.10 ACI_GATT_SET_EVENT_MASK

Description

Mask events from the GATT. The default configuration is all the events masked.

Input parameters

Table 234. **ACI_GATT_SET_EVENT_MASK input parameters**

Parameter	Size	Description	Possible values
GATT_Evt_Mask	4	GATT/ATT event mask.	• 0x00000001: ACI_GATT_ATTRIBUTE_MODIFIED_EVENT • 0x00000002: ACI_GATT_PROC_TIMEOUT_EVENT • 0x00000004: ACI_ATT_EXCHANGE_MTU_RESP_EVENT • 0x00000008: ACI_ATT_FIND_INFO_RESP_EVENT • 0x00000010: ACI_ATT_FIND_BY_TYPE_VALUE_RESP_EVENT • 0x00000020: ACI_ATT_READ_BY_TYPE_RESP_EVENT • 0x00000040: ACI_ATT_READ_RESP_EVENT • 0x00000080: ACI_ATT_READ_BLOB_RESP_EVENT • 0x00000100: ACI_ATT_READ_MULTIPLE_RESP_EVENT • 0x00000200: ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT • 0x00000800: ACI_ATT_PREPARE_WRITE_RESP_EVENT • 0x00001000: ACI_ATT_EXEC_WRITE_RESP_EVENT • 0x00002000: ACI_GATT_INDICATION_EVENT • 0x00004000: ACI_GATT_NOTIFICATION_EVENT • 0x00008000: ACI_GATT_ERROR_RESP_EVENT • 0x00010000: ACI_GATT_PROC_COMPLETE_EVENT • 0x00020000: ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT • 0x00040000: ACI_GATT_TX_POOL_AVAILABLE_EVENT • 0x00100000: ACI_GATT_READ_EXT_EVENT • 0x00200000: ACI_GATT_INDICATION_EXT_EVENT • 0x00400000: ACI_GATT_NOTIFICATION_EXT_EVENT

Output parameters

Table 235. **ACI_GATT_SET_EVENT_MASK output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.11 ACI_GATT_EXCHANGE_CONFIG

Description

Perform an ATT MTU exchange procedure. When the ATT MTU exchange procedure is completed, a ACI_ATT_EXCHANGE_MTU_RESP_EVENT event is generated. A ACI_GATT_PROC_COMPLETE_EVENT event is also generated to indicate the end of the procedure.

Input parameters

Table 236. **ACI_GATT_EXCHANGE_CONFIG input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 237. **ACI_GATT_EXCHANGE_CONFIG output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_EXCHANGE_MTU_RESP_EVENT

2.5.12 ACI_ATT_FIND_INFO_REQ

Description

Send a find information request. This command is used to obtain the mapping of attribute handles with their associated types. The responses of the procedure are given through the ACI_ATT_FIND_INFO_RESP_EVENT event. The end of the procedure is indicated by a ACI_GATT_PROC_COMPLETE_EVENT event.

Input parameters

Table 238. **ACI_ATT_FIND_INFO_REQ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Start_Handle	2	First requested handle number	-
End_Handle	2	Last requested handle number	-

Output parameters

Table 239. **ACI_ATT_FIND_INFO_REQ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_FIND_INFO_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.13 ACI_ATT_FIND_BY_TYPE_VALUE_REQ

Description

Send a find by type value request. The find by type value request is used to obtain the handles of attributes that have a given 16-bit UUID attribute type and a given attribute value. The responses of the procedure are given through the ACI_ATT_FIND_BY_TYPE_VALUE_RESP_EVENT event. The end of the procedure is indicated by a ACI_GATT_PROC_COMPLETE_EVENT event.

Input parameters

Table 240. **ACI_ATT_FIND_BY_TYPE_VALUE_REQ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Start_Handle	2	First requested handle number	-
End_Handle	2	Last requested handle number	-
UUID	2	2 octet UUID to find (little-endian)	-
Attribute_Val_Length	1	Length of attribute value (maximum value is ATT_MTU - 7).	-
Attribute_Val	Attribute_Val_Length	Attribute value to find	-

Output parameters

Table 241. **ACI_ATT_FIND_BY_TYPE_VALUE_REQ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_FIND_INFO_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.14 ACI_ATT_READ_BY_TYPE_REQ

Description

Send a read by type request. The read by type request is used to obtain the values of attributes where the attribute type is known but the handle is not known. The responses are given through the ACI_ATT_READ_BY_TYPE_RESP_EVENT event.

Input parameters

Table 242. **ACI_ATT_READ_BY_TYPE_REQ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Start_Handle	2	First requested handle number	-
End_Handle	2	Last requested handle number	-
UUID_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
UUID	2 or 16	16-bit UUID or 128-bit UUID	-

Output parameters

Table 243. **ACI_ATT_READ_BY_TYPE_REQ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_READ_BY_TYPE_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.15 ACI_ATT_READ_BY_GROUP_TYPE_REQ

Description

Send a read by group type request. The read by group type request is used to obtain the values of grouping attributes where the attribute type is known but the handle is not known. Grouping attributes are defined at GATT layer. The grouping attribute types are: "Primary Service", "Secondary Service" and "Characteristic". The responses of the procedure are given through the ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT event. The end of the procedure is indicated by a ACI_GATT_PROC_COMPLETE_EVENT.

Input parameters

Table 244. **ACI_ATT_READ_BY_GROUP_TYPE_REQ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Start_Handle	2	First requested handle number	-
End_Handle	2	Last requested handle number	-
UUID_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
UUID	2 or 16	16-bit UUID or 128-bit UUID	-

Output parameters

Table 245. **ACI_ATT_READ_BY_GROUP_TYPE_REQ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.16 ACI_ATT_PREPARE_WRITE_REQ

Description

Send a prepare write request. The prepare write request is used to request the server to prepare to write the value of an attribute. The responses of the procedure are given through the ACI_ATT_PREPARE_WRITE_RESP_EVENT event. The end of the procedure is indicated by a ACI_GATT_PROC_COMPLETE_EVENT.

Input parameters

Table 246. **ACI_ATT_PREPARE_WRITE_REQ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the attribute to be written	-
Val_Offset	2	The offset of the first octet to be written	-
Attribute_Val_Length	1	Length of attribute value (maximum value is ATT_MTU - 5).	-
Attribute_Val	Attribute_Val_Length	The value of the attribute to be written	-

Output parameters

Table 247. **ACI_ATT_PREPARE_WRITE_REQ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_PREPARE_WRITE_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.17 ACI_ATT_EXECUTE_WRITE_REQ

Description

Send an execute write request. The execute write request is used to request the server to write or cancel the write of all the prepared values currently held in the prepare queue from this client. The result of the procedure is given through the ACI_ATT_EXEC_WRITE_RESP_EVENT event. The end of the procedure is indicated by a ACI_GATT_PROC_COMPLETE_EVENT event.

Input parameters

Table 248. **ACI_ATT_EXECUTE_WRITE_REQ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Execute	1	Execute or cancel writes.	• 0x00: Cancel all prepared writes • 0x01: Immediately write all pending prepared values

Output parameters

Table 249. **ACI_ATT_EXECUTE_WRITE_REQ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_EXEC_WRITE_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.18 ACI_GATT_DISC_ALL_PRIMARY_SERVICES

Description

Start the GATT client procedure to discover all primary services on the server. The responses of the procedure are given through the ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT event.

Input parameters

Table 250. ACI_GATT_DISC_ALL_PRIMARY_SERVICES input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 251. ACI_GATT_DISC_ALL_PRIMARY_SERVICES output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.19 ACI_GATT_DISC_PRIMARY_SERVICE_BY_UUID

Description

Start the procedure to discover the primary services of the specified UUID on the server. The responses of the procedure are given through the ACI_ATT_FIND_BY_TYPE_VALUE_RESP_EVENT event. The end of the procedure is indicated by a ACI_GATT_PROC_COMPLETE_EVENT event.

Input parameters

Table 252. ACI_GATT_DISC_PRIMARY_SERVICE_BY_UUID input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
UUID_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
UUID	2 or 16	16-bit UUID or 128-bit UUID	-

Output parameters

Table 253. ACI_GATT_DISC_PRIMARY_SERVICE_BY_UUID output parameters

Parameter	Size	Description	Possible values
Status	1	Error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- ACI_ATT_FIND_BY_TYPE_VALUE_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT

- ACI_GATT_PROC_COMPLETE_EVENT

2.5.20 ACI_GATT_FIND_INCLUDED_SERVICES

Description

Start the procedure to find all included services. The responses of the procedure are given through the ACI_ATT_READ_BY_TYPE_RESP_EVENT event. The end of the procedure is indicated by a ACI_GATT_PROC_COMPLETE_EVENT event.

Input parameters

Table 254. ACI_GATT_FIND_INCLUDED_SERVICES input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Start_Handle	2	Start attribute handle of the service	-
End_Handle	2	End attribute handle of the service	-

Output parameters

Table 255. ACI_GATT_FIND_INCLUDED_SERVICES output parameters

Parameter	Size	Description	Possible values
Status	1	Error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.21 ACI_GATT_DISC_ALL_CHAR_OF_SERVICE

Description

Start the procedure to discover all the characteristics of a given service. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. Before procedure completion the response packets are given through ACI_ATT_READ_BY_TYPE_RESP_EVENT event.

Input parameters

Table 256. ACI_GATT_DISC_ALL_CHAR_OF_SERVICE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Start_Handle	2	Start attribute handle of the service	-
End_Handle	2	End attribute handle of the service	-

Output parameters

Table 257. ACI_GATT_DISC_ALL_CHAR_OF_SERVICE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.22 ACI_GATT_DISC_CHAR_BY_UUID

Description

Start the procedure to discover all the characteristics specified by a UUID. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. Before procedure completion the response packets are given through ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT event.

Input parameters

Table 258. ACI_GATT_DISC_CHAR_BY_UUID input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Start_Handle	2	Start attribute handle of the service	-
End_Handle	2	End attribute handle of the service	-
UUID_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
UUID	2 or 16	16-bit UUID or 128-bit UUID	-

Output parameters

Table 259. ACI_GATT_DISC_CHAR_BY_UUID output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.23 ACI_GATT_DISC_ALL_CHAR_DESC

Description

Start the procedure to discover all characteristic descriptors on the server. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. Before procedure completion the response packets are given through ACI_ATT_FIND_INFO_RESP_EVENT event.

Input parameters

Table 260. ACI_GATT_DISC_ALL_CHAR_DESC input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Char_Handle	2	Handle of the characteristic value	-
End_Handle	2	End handle of the characteristic	-

Output parameters

Table 261. **ACI_GATT_DISC_ALL_CHAR_DESC output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_FIND_INFO_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.24 ACI_GATT_READ_CHAR_VALUE

Description

Start the procedure to read the attribute value. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. Before procedure completion the response packet is given through ACI_ATT_READ_RESP_EVENT event.

Input parameters

Table 262. **ACI_GATT_READ_CHAR_VALUE input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the characteristic value to be read	-

Output parameters

Table 263. **ACI_GATT_READ_CHAR_VALUE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_READ_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.25 ACI_GATT_READ_USING_CHAR_UUID

Description

This command sends a Read By Type Request packet to the server in order to read the value attribute of the characteristics specified by the UUID. When the procedure is completed, an ACI_GATT_PROC_COMPLETE_EVENT event is generated. Before procedure completion the response packets are given through ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT event.

Note: The number of bytes of a value reported by ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT event cannot exceed BLE_EVT_MAX_PARAM_LEN - 7 (i.e. 248 bytes for default value of BLE_EVT_MAX_PARAM_LEN).

Input parameters

Table 264. **ACI_GATT_READ_USING_CHAR_UUID input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Start_Handle	2	Starting handle of the range to be searched	-
End_Handle	2	End handle of the range to be searched	-
UUID_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
UUID	2 or 16	16-bit UUID or 128-bit UUID	-

Output parameters

Table 265. **ACI_GATT_READ_USING_CHAR_UUID output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.26 ACI_GATT_READ_LONG_CHAR_VALUE

Description

Start the procedure to read a long characteristic value. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. Before procedure completion the response packets are given through ACI_ATT_READ_BLOB_RESP_EVENT event.

Input parameters

Table 266. **ACI_GATT_READ_LONG_CHAR_VALUE input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the characteristic value to be read	-
Val_Offset	2	Offset from which the value needs to be read	-

Output parameters

Table 267. **ACI_GATT_READ_LONG_CHAR_VALUE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_READ_BLOB_RESP_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.27 ACI_GATT_READ_MULTIPLE_CHAR_VALUE

Description

Start a procedure to read multiple characteristic values from a server. This sub-procedure is used to read multiple characteristic values from a server when the client knows the characteristic value handles. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. Before procedure completion the response packets are given through ACI_ATT_READ_MULTIPLE_RESP_EVENT event.

Input parameters

Table 268. ACI_GATT_READ_MULTIPLE_CHAR_VALUE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Number_of_Handles	1	The number of handles for which the value has to be read	-
Handle[i]	2	The handles for which the attribute value has to be read	-

Output parameters

Table 269. ACI_GATT_READ_MULTIPLE_CHAR_VALUE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GATT_ERROR_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.28 ACI_GATT_WRITE_CHAR_VALUE

Description

Start the procedure to write a characteristic value. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated.

Input parameters

Table 270. ACI_GATT_WRITE_CHAR_VALUE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the characteristic value to be written	-
Attribute_Val_Length	1	Length of the value to be written	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Output parameters

Table 271. ACI_GATT_WRITE_CHAR_VALUE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT

- ACI_GATT_PROC_COMPLETE_EVENT
- ACI_GATT_ERROR_RESP_EVENT

2.5.29 ACI_GATT_WRITE_LONG_CHAR_VALUE

Description

Start the procedure to write a long characteristic value. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. During the procedure, ACI_ATT_PREPARE_WRITE_RESP_EVENT and ACI_ATT_EXEC_WRITE_RESP_EVENT events are raised.

Input parameters

Table 272. ACI_GATT_WRITE_LONG_CHAR_VALUE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the characteristic value to be written	-
Val_Offset	2	Offset at which the attribute has to be written	-
Attribute_Val_Length	1	Length of the value to be written	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Output parameters

Table 273. ACI_GATT_WRITE_LONG_CHAR_VALUE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT
- ACI_ATT_PREPARE_WRITE_RESP_EVENTACI_ATT_EXEC_WRITE_RESP_EVENT
-

2.5.30 ACI_GATT_WRITE_CHAR_RELIABLE

Description

Start the procedure to write a characteristic reliably. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. During the procedure, ACI_ATT_PREPARE_WRITE_RESP_EVENT and ACI_ATT_EXEC_WRITE_RESP_EVENT events are raised.

Input parameters

Table 274. ACI_GATT_WRITE_CHAR_RELIABLE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the attribute to be written	-
Val_Offset	2	Offset at which the attribute has to be written	-
Attribute_Val_Length	1	Length of the value to be written	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Output parameters

Table 275. **ACI_GATT_WRITE_CHAR_RELIABLE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT
- ACI_ATT_PREPARE_WRITE_RESP_EVENT
- ACI_ATT_EXEC_WRITE_RESP_EVENT

2.5.31 ACI_GATT_WRITE_LONG_CHAR_DESC

Description

Start the procedure to write a long characteristic descriptor. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. During the procedure, ACI_ATT_PREPARE_WRITE_RESP_EVENT and ACI_ATT_EXEC_WRITE_RESP_EVENT events are raised.

Input parameters

Table 276. **ACI_GATT_WRITE_LONG_CHAR_DESC input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the attribute to be written	-
Val_Offset	2	Offset at which the attribute has to be written	-
Attribute_Val_Length	1	Length of the value to be written	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Output parameters

Table 277. **ACI_GATT_WRITE_LONG_CHAR_DESC output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT
- ACI_ATT_PREPARE_WRITE_RESP_EVENT
- ACI_ATT_EXEC_WRITE_RESP_EVENT

2.5.32 ACI_GATT_READ_LONG_CHAR_DESC

Description

Start the procedure to read a long characteristic value. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. Before procedure completion the response packets are given through ACI_ATT_READ_BLOB_RESP_EVENT event.

Input parameters

Table 278. **ACI_GATT_READ_LONG_CHAR_DESC input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the characteristic descriptor	-
Val_Offset	2	Offset from which the value needs to be read	-

Output parameters

Table 279. **ACI_GATT_READ_LONG_CHAR_DESC output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_READ_BLOB_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.33 ACI_GATT_WRITE_CHAR_DESC

Description

Start the procedure to write a characteristic descriptor. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated.

Input parameters

Table 280. **ACI_GATT_WRITE_CHAR_DESC input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the attribute to be written	-
Attribute_Val_Length	1	Length of the value to be written	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Output parameters

Table 281. **ACI_GATT_WRITE_CHAR_DESC output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.34 ACI_GATT_READ_CHAR_DESC

Description

Start the procedure to read the descriptor specified. When the procedure is completed, a ACI_GATT_PROC_COMPLETE_EVENT event is generated. Before procedure completion the response packet is given through ACI_ATT_READ_RESP_EVENT event.

Input parameters

Table 282. **ACI_GATT_READ_CHAR_DESC input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the descriptor to be read	-

Output parameters

Table 283. **ACI_GATT_READ_CHAR_DESC output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_ATT_READ_RESP_EVENT
- ACI_GATT_PROC_COMPLETE_EVENT

2.5.35 ACI_GATT_WRITE_WITHOUT_RESP

Description

Start the procedure to write a characteristic value without waiting for any response from the server. No events are generated after this command is executed.

The length of the value to be written must not exceed (ATT_MTU - 3); it must also not exceed (BLE_EVT_MAX_PARAM_LEN - 5) i.e. 250 for BLE_EVT_MAX_PARAM_LEN default value.

Input parameters

Table 284. **ACI_GATT_WRITE_WITHOUT_RESP input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the characteristic value to be written	-
Attribute_Val_Length	1	Length of the value to be written	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Output parameters

Table 285. **ACI_GATT_WRITE_WITHOUT_RESP output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.36 ACI_GATT_SIGNED_WRITE_WITHOUT_RESP

Description

Start a signed write without response from the server. The procedure is used to write a characteristic value with an authentication signature without waiting for any response from the server. It cannot be used when the link is encrypted.

The length of the value to be written must not exceed (ATT_MTU - 15); it must also not exceed (BLE_EVT_MAX_PARAM_LEN - 5) i.e. 250 for BLE_EVT_MAX_PARAM_LEN default value.

Input parameters

Table 286. **ACI_GATT_SIGNED_WRITE_WITHOUT_RESP input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the characteristic value to be written	-
Attribute_Val_Length	1	Length of the value to be written	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Output parameters

Table 287. **ACI_GATT_SIGNED_WRITE_WITHOUT_RESP output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.37 ACI_GATT_CONFIRM_INDICATION

Description

Allow application to confirm indication. This command has to be sent when the application receives the event ACI_GATT_INDICATION_EVENT.

Input parameters

Table 288. **ACI_GATT_CONFIRM_INDICATION input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 289. **ACI_GATT_CONFIRM_INDICATION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.38 ACI_GATT_WRITE_RESP

Description

Allow or reject a write request from a client. This command has to be sent by the application when it receives the ACI_GATT_WRITE_PERMIT_REQ_EVENT. If the write can be allowed, then the status and error code has to be set to 0. If the write cannot be allowed, then the status has to be set to 1 and the error code has to be set to the error code that has to be passed to the client.

Input parameters

Table 290. **ACI_GATT_WRITE_RESP input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Attr_Handle	2	Handle of the attribute that was passed in the event EVT_BLUE_GATT_WRITE_PERMIT_REQ	-
Write_status	1	If the value can be written or not.	• 0x00: The value can be written to the attribute specified by attr_handle • 0x01: The value cannot be written to the attribute specified by the attr_handle
Error_Code	1	The error code that has to be passed to the client in case the write has to be rejected	-
Attribute_Val_Length	1	Length of the value to be written as passed in the event EVT_BLUE_GATT_WRITE_PERMIT_REQ	-
Attribute_Val	Attribute_Val_Length	Value as passed in the event EVT_BLUE_GATT_WRITE_PERMIT_REQ	-

Output parameters

Table 291. **ACI_GATT_WRITE_RESP output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.39 ACI_GATT_ALLOW_READ

Description

Allow the GATT server to send a response to a read request from a client. The application has to send this command when it receives the ACI_GATT_READ_PERMIT_REQ_EVENT or ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT. This command indicates to the stack that the response can be sent to the client. So if the application wishes to update any of the attributes before they are read by the client, it has to update the characteristic values using the ACI_GATT_UPDATE_CHAR_VALUE and then give this command. The application must perform the required operations within 30 seconds. Otherwise the GATT procedure is timeout.

Input parameters

Table 292. **ACI_GATT_ALLOW_READ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF

Output parameters

Table 293. **ACI_GATT_ALLOW_READ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.40 ACI_GATT_SET_SECURITY_PERMISSION

Description

This command sets the security permission for the attribute handle specified. Currently the setting of security permission is allowed only for client configuration descriptor.

Input parameters

Table 294. **ACI_GATT_SET_SECURITY_PERMISSION input parameters**

Parameter	Size	Description	Possible values
Serv_Handle	2	Handle of the service which contains the attribute whose security permission has to be modified	-
Attr_Handle	2	Handle of the attribute whose security permission has to be modified	-
Security_Permissions	1	Security permission flags.	Bitmask of: • 0x00: None • 0x01: AUTHEN_READ (Need authentication to read) • 0x02: AUTHOR_READ (Need authorization to read) • 0x04: ENCRY_READ (Need encryption to read) • 0x08: AUTHEN_WRITE (need authentication to write) • 0x10: AUTHOR_WRITE (need authorization to write) • 0x20: ENCRY_WRITE (need encryption to write)

Output parameters

Table 295. **ACI_GATT_SET_SECURITY_PERMISSION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.41 ACI_GATT_SET_DESC_VALUE

Description

This command sets the value of the descriptor specified by charDescHandle.

Input parameters

Table 296. **ACI_GATT_SET_DESC_VALUE input parameters**

Parameter	Size	Description	Possible values
Serv_Handle	2	Handle of the service which contains the characteristic descriptor	-
Char_Handle	2	Handle of the characteristic which contains the descriptor	-
Char_Desc_Handle	2	Handle of the descriptor whose value has to be set	-
Val_Offset	2	Offset from which the descriptor value has to be updated	-
Char_Desc_Value_Length	1	Length of the descriptor value	-
Char_Desc_Value	Char_Desc_Value_Length	Descriptor value	-

Output parameters

Table 297. **ACI_GATT_SET_DESC_VALUE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.42 ACI_GATT_READ_HANDLE_VALUE

Description

Reads the value of the attribute handle specified from the local GATT database.

Input parameters

Table 298. **ACI_GATT_READ_HANDLE_VALUE input parameters**

Parameter	Size	Description	Possible values
Attr_Handle	2	Handle of the attribute to read	-
Offset	2	Offset from which the value needs to be read	-
Value_Length_Requested	2	Maximum number of octets to be returned as attribute value	-

Output parameters

Table 299. **ACI_GATT_READ_HANDLE_VALUE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Length	2	Length of the attribute value	-
Value_Length	2	Length in octets of the Value parameter	-
Value	Value_Length	Attribute value	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.43 ACI_GATT_UPDATE_CHAR_VALUE_EXT

Description

This command is a more flexible version of ACI_GATT_UPDATE_CHAR_VALUE to support update of long attribute up to 512 bytes and indicate selectively the generation of indication/notification.

Input parameters

Table 300. **ACI_GATT_UPDATE_CHAR_VALUE_EXT input parameters**

Parameter	Size	Description	Possible values
Conn_Handle_To_Notify	2	Connection handle to notify. Notify all subscribed clients if equal to 0x0000	-
Service_Handle	2	Handle of service to which the characteristic belongs	-
Char_Handle	2	Handle of the characteristic declaration	-
Update_Type	1	Allow notification or Indication generation, if enabled in the client characteristic configuration descriptor.	Bitmask of: • 0x00: Do not notify • 0x01: Notification • 0x02: Indication
Char_Length	2	Total length of the characteristic value. In case of a variable size characteristic, this field specifies the new length of the characteristic value after the update; in case of fixed length characteristic this field is ignored.	-
Value_Offset	2	The offset from which the attribute value has to be updated.	-
Value_Length	1	Length of the value parameter in octets	-
Value	Value_Length	Updated characteristic value	-

Output parameters

Table 301. **ACI_GATT_UPDATE_CHAR_VALUE_EXT output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.44 ACI_GATT_DENY_READ

Description

Deny the GATT server to send a response to a read request from a client. The application may send this command when it receives the ACI_GATT_READ_PERMIT_REQ_EVENT or ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT. This command indicates to the stack that the client is not allowed to read the requested characteristic due to e.g. application restrictions. The error code is either 0x08 (insufficient authorization) or a value in the range 0x80-0x9F (application error). The application must issue the ACI_GATT_DENY_READ or ACI_GATT_ALLOW_READ command within 30 seconds from the reception of the ACI_GATT_READ_PERMIT_REQ_EVENT or ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT events otherwise the GATT procedure issues a timeout.

Input parameters

Table 302. **ACI_GATT_DENY_READ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Error_Code	1	Error code for the command	• 0x08: Insufficient authorization • 0x80 ... 0x9F: Application error

Output parameters

Table 303. **ACI_GATT_DENY_READ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.45 ACI_GATT_SET_ACCESS_PERMISSION

Description

This command sets the access permission for the attribute handle specified.

Input parameters

Table 304. **ACI_GATT_SET_ACCESS_PERMISSION input parameters**

Parameter	Size	Description	Possible values
Serv_Handle	2	Handle of the service which contains the attribute whose access permission has to be modified	-
Attr_Handle	2	Handle of the attribute whose security permission has to be modified	-
Access_Permissions	1	Access permission	Bitmask of: • 0x00: None • 0x01: READ • 0x02: WRITE • 0x04: WRITE_WO_RESP • 0x08: SIGNED_WRITE

Output parameters

Table 305. **ACI_GATT_SET_ACCESS_PERMISSION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.6 L2CAP commands

In Table 306 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / SO, respectively, for Link layer only / Beacon only / Slave only.

Table 306. L2CAP commands list

Command	OpCode	LO	SO	BO
ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_REQ	0xFD81	-	Y	-
ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_RESP	0xFD82	-	-	-

2.6.1 ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_REQ

Description

Send an L2CAP connection parameter update request from the slave to the master. An ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT event is raised when the master responds to the request (accepts or rejects).

Input parameters

Table 307. ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_REQ input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Conn_Interval_Min	2	Minimum value for the connection event interval. This is less than or equal to Conn_Interval_Max. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Interval_Max	2	Maximum value for the connection event interval. This is greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Slave_latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Timeout_Multiplier	2	Defines connection timeout parameter in the following manner: Timeout Multiplier * 10ms.	-

Output parameters

Table 308. ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_REQ output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT

2.6.2 ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_RESP

Description

Accept or reject a connection update. This command must be sent in response to a ACI_L2CAP_CONNECTION_UPDATE_REQ_EVENT event from the controller. The accept parameter has to be set if the connection parameters given in the event are acceptable.

Input parameters

Table 309. **ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_RESP input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Conn_Interval_Min	2	Minimum value for the connection event interval. This is less than or equal to Conn_Interval_Max. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Interval_Max	2	Maximum value for the connection event interval. This is greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Slave_latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Timeout_Multiplier	2	Defines connection timeout parameter in the following manner: Timeout Multiplier * 10ms.	-
Minimum_CE_Length	2	Information parameter about the minimum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Maximum_CE_Length	2	Information parameter about the maximum length of connection needed for this LE connection. Time = N * 0.625 ms	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Identifier	1	Identifier received in ACI_L2CAP_Connection_Update_Req event.	-
Accept	1	Specify if connection update parameters are acceptable or not.	• 0x00: Reject • 0x01: Accept

Output parameters

Table 310. **ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_RESP output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

3 ACI/HCI events

3.1 HCI events

In Table 311 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / SO, respectively, for Link layer only / Beacon only / Slave only.

Table 311. HCI events commands list

Command	OpCode	LO	SO	BO
HCI_DISCONNECTION_COMPLETE_EVENT	0x05	Y	Y	-
HCI_ENCRYPTION_CHANGE_EVENT	0x08	Y	Y	-
HCI_READ_REMOTE_VERSION_INFORMATION_COMPLETE_EVENT	0x0C	Y	Y	-
HCI_HARDWARE_ERROR_EVENT	0x10	Y	Y	Y
HCI_NUMBER_OF_COMPLETED_PACKETS_EVENT	0x13	Y	-	-
HCI_ENCRYPTION_KEY_REFRESH_COMPLETE_EVENT	0x30	Y	Y	-
HCI_COMMAND_COMPLETE_EVENT	0x0E	Y	Y	Y
HCI_COMMAND_STATUS_EVENT	0x0F	Y	Y	Y

3.1.1 HCI_DISCONNECTION_COMPLETE_EVENT

Description

The disconnection complete event occurs when a connection is terminated. The status parameter indicates if the disconnection was successful or not. The reason parameter indicates the reason for the disconnection if the disconnection was successful. If the disconnection was not successful, the value of the reason parameter can be ignored by the host. For example, this can be the case if the host has issued the disconnect command and there was a parameter error, or the command was not presently allowed, or a Connection_Handle that didn't correspond to a connection was given.

Event parameters

Table 312. HCI_DISCONNECTION_COMPLETE_EVENT event parameters

Parameter	Size	Description	Possible values
Status	1	Error code	-
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Reason	1	Reason for disconnection (see Bluetooth Core Specification [Vol 2] Part D, Error Codes)	-

3.1.2 HCI_ENCRYPTION_CHANGE_EVENT

Description

The Encryption Change event is used to indicate that the change of the encryption mode has been completed. The Connection_Handle is a Connection_Handle for an ACL connection. The Encryption_Enabled event parameter specifies the new Encryption_Enabled parameter for the Connection_Handle specified by the Connection_Handle event parameter. This event occurs on both devices to notify the hosts when encryption has changed for the specified Connection_Handle between two device. The Encryption change event is used to indicate that the change of the encryption mode has been completed. The Connection_Handle is a Connection_Handle for an ACL connection. The Encryption_Enabled event parameter specifies the new Encryption_Enabled parameter for the Connection_Handle specified by the Connection_Handle event parameter. This event occurs on both devices to notify the hosts when encryption has changed for the specified Connection_Handle between two devices.

Note: This event is not be generated if encryption is paused or resumed; during a role switch, for example.

The meaning of the Encryption_Enabled parameter depends on whether the host has indicated support for secure connections in the Secure_Connections_Host_Support parameter. When Secure_Connections_Host_Support is 'disabled' or the Connection_Handle refers to an LE link, the controller only use Encryption_Enabled values 0x00 (OFF) and 0x01 (ON).

Event parameters

Table 313. HCI_ENCRYPTION_CHANGE_EVENT event parameters

Parameter	Size	Description	Possible values
Status	1	Error code	-
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Encryption_Enabled	1	Link level encryption	• 0x00: Link level encryption OFF • 0x01: Link level encryption is ON with AES-CCM

3.1.3 HCI_READ_REMOTE_VERSION_INFORMATION_COMPLETE_EVENT

Description

The read remote version information complete event is used to indicate the completion of the process obtaining the version information of the remote controller specified by the Connection_Handle event parameter.

The Connection_Handle is for an ACL connection. The version event parameter defines the specification version of the LE controller. The Manufacturer_Name event parameter indicates the manufacturer of the remote controller. The subversion event parameter is controlled by the manufacturer and is implementation dependent. The subversion event parameter defines the various revisions that each version of the Bluetooth hardware goes through as design processes change and errors are fixed. This allows the software to determine what Bluetooth hardware is being used and, if necessary, to work around various bugs in the hardware. When the Connection_Handle is associated with an LE-U logical link, the version event parameter is link layer VersNr parameter, the Manufacturer_Name event parameter is the compId parameter, and the subversion event parameter is the SubVersNr parameter.

Event parameters

Table 314. HCI_READ_REMOTE_VERSION_INFORMATION_COMPLETE_EVENT event parameters

Parameter	Size	Description	Possible values
Status	1	Error code	-
Connection_Handle	2	Connection handle for which the command is given	0x0000 ... 0x0EFF
Version	1	Version of the current LMP in the remote controller	-
Manufacturer_Name	2	Manufacturer Name of the remote Controller	-
Subversion	2	Subversion of the LMP in the remote Controller	-

3.1.4 HCI_HARDWARE_ERROR_EVENT

Description

The hardware error event is used to indicate some implementation specific type of hardware failure for the controller. This event is used to notify the host that a hardware failure has occurred in the controller.

Event parameters

Table 315. HCI_HARDWARE_ERROR_EVENT event parameters

Parameter	Size	Description	Possible values
Hardware_Code	1	Hardware error event code. Error code 0 is not used. Error code is bluecore act2 error detected. Error code 2 is bluecore time overrun error detected. Error code 3 is internal FIFO full.	• 0x00: Not used • 0x01: event_act2 error • 0x02: event_time_overrun error • 0x03: event_fifo_full error

3.1.5 HCI_NUMBER_OF_COMPLETED_PACKETS_EVENT

Description

The number of completed packets event is used by the controller to indicate to the host how many HCI data packets have been completed (transmitted or flushed) for each Connection_Handle since the previous number of completed packets event was sent to the host. This means that the corresponding buffer space has been freed in the controller. Based on this information, and the HC_Total_Num_ACL_Data_Packets and HC_Total_Num_Synchronous_- Data_Packets return parameter of the Read_Buffer_Size command, the host determines for which Connection_Handles the following HCI data packets must be sent to the controller. The number of completed packets event must not be sent before the corresponding connection complete event. While the controller has HCI data packets in its buffer, it must keep sending the number of completed packets event to the host at least periodically, until it finally reports that all the pending ACL data packets have been transmitted or flushed.

Event parameters

Table 316. HCI_NUMBER_OF_COMPLETED_PACKETS_EVENT event parameters

Parameter	Size	Description	Possible values
Number_of_Handles	1	The number of Connection_Handles and Num_HCI_Data_Packets parameters pairs contained in this event	-
Connection_Handle[i]	2	Connection handle	-
HC_Num_Of_Completed_Packets[i]	2	The number of HCI data packets that have been completed (transmitted or flushed) for the associated Connection_Handle since the previous time the event was returned	-

3.1.6 HCI_ENCRYPTION_KEY_REFRESH_COMPLETE_EVENT

Description

The encryption key refresh complete event is used to indicate to the host that the encryption key was refreshed on the given Connection_Handle any time encryption is paused and then resumed. If the encryption key refresh complete event was generated due to an encryption pause and resume operation embedded within a change connection link key procedure, the encryption key refresh complete event is sent prior to the change connection link key complete event. If the encryption key refresh complete event was generated due to an encryption pause and resume operation embedded within a role switch procedure, the encryption key refresh complete event is sent prior to the role change event.

Event parameters

Table 317. **HCI_ENCRYPTION_KEY_REFRESH_COMPLETE_EVENT event parameters**

Parameter	Size	Description	Possible values
Status	1	Error code	-
Connection_Handle	2	Connection handle for which the command is given	0x0000 ... 0x0EFF

3.1.7 HCI_COMMAND_COMPLETE_EVENT

Description

The command complete event is used by the controller for most commands to transmit return status of a command and the other event parameters that are specified for the issued HCI command. The Num_HCI_Command_Packets event parameter allows the controller to indicate the number of HCI command packets the host can send to the controller. If the controller requires the host to stop sending commands, the Num_HCI_Command_Packets event parameter is set to zero. To indicate to the host that the controller is ready to receive HCI command packets, the controller generates a command complete event with the Command_Opcode 0x0000, and the Num_HCI_Command_Packets event parameter is set to 1 or more. See each command for the parameters that are returned by this event.

Event parameters

Table 318. **HCI_COMMAND_COMPLETE_EVENT event parameters**

Parameter	Size	Description	Possible values
Num_HCI_Command_Packets	1	The Number of HCI command packets which are allowed to be sent to the controller from the host.	-
Command_Opcode	2	Opcode of the command which caused this event	-
Return_Parameters	Variable	This is the return parameter(s) for the command specified in the Command_Opcode event parameter. See each command's definition for the list of return parameters associated with that command.	-

3.1.8 HCI_COMMAND_STATUS_EVENT

Description

The command status event is used to indicate that the command described by the Command_Opcode parameter has been received, and that the controller is currently performing the task for this command. This event is needed to provide mechanisms for asynchronous operation, which makes it possible to prevent the host from waiting for a command to finish. If the command cannot begin to execute (a parameter error may have occurred, or the command may currently not be allowed), the status event parameter contains the corresponding error code, and no complete event follows since the command was not started.

The Num_HCI_Command_Packets event parameter allows the controller to indicate the number of HCI command packets the host can send to the controller. If the controller requires the host to stop sending commands, the Num_HCI_Command_Packets event parameter is set to zero. To indicate to the host that the controller is ready to receive HCI command packets, the controller generates a command status event with status 0x00 and Command_Opcode 0x0000, and the Num_HCI_Command_Packets event parameter is set to 1 or more.

Event parameters

Table 319. **HCI_COMMAND_STATUS_EVENT event parameters**

Parameter	Size	Description	Possible values
Status	1	Error code	-
Num_HCI_Command_Packets	1	The Number of HCI command packets which are allowed to be sent to the controller from the Host	-
Command_Opcode	2	Opcode of the command which caused this event	-

3.2 HCI LE META events

In Table 320 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO /BO / SO, respectively, for Link layer only / Beacon only / Slave only.

Table 320. HCI LE META events commands list

Command	OpCode	LO	SO	BO
HCI_LE_CONNECTION_COMPLETE_EVENT	0x01	Y	Y	-
HCI_LE_ADVERTISING_REPORT_EVENT	0x02	Y	-	Y
HCI_LE_CONNECTION_UPDATE_COMPLETE_EVENT	0x03	Y	Y	-
HCI_LE_READ_REMOTE_FEATURES_COMPLETE_EVENT	0x04	Y	Y	-
HCI_LE_LONG_TERM_KEY_REQUEST_EVENT	0x05	Y	Y	-
HCI_LE_DATA_LENGTH_CHANGE_EVENT	0x07	Y	Y	-
HCI_LE_READ_LOCAL_P256_PUBLIC_KEY_COMPLETE_EVENT	0x08	Y	Y	-
HCI_LE_GENERATE_DHKEY_COMPLETE_EVENT	0x09	Y	Y	-
HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT	0x0A	Y	Y	-
HCI_LE_DIRECT_ADVERTISING_REPORT_EVENT	0x0B	Y	-	Y
HCI_LE_PHY_UPDATE_COMPLETE_EVENT	0x0C	Y	-	-

3.2.1 HCI_LE_CONNECTION_COMPLETE_EVENT

Description

The LE connection complete event indicates to both of the hosts forming the connection that a new connection has been created. Upon the creation of the connection a Connection_Handle is assigned by the controller, and passed to the host in this event. If the connection establishment fails this event is provided to the host that had issued the LE_Create_Connection command. This event indicates to the host which issued a LE_Create_Connection command and received a command status event if the connection establishment failed or was successful. The Master_Clock_Accuracy parameter is only valid for a slave. On a master, this parameter is set to 0x00.

Event parameters

Table 321. HCI_LE_CONNECTION_COMPLETE_EVENT event parameters

Parameter	Size	Description	Possible values
Status	1	Error code	-
Connection_Handle	2	Connection handle to be used to identify the connection with the peer device	0x0000 ... 0x0EFF
Role	1	Role of the local device in the connection	• 0x00: Master • 0x01: Slave
Peer_Address_Type	1	The address type of the peer device	• 0x00: Public device address • 0x01: Random device address
Peer_Address	6	Public device address or random device address of the peer device	-
Conn_Interval	2	Connection interval used on this connection. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3

Parameter	Size	Description	Possible values
Supervision_Timeout	2	Supervision timeout for the LE link. It is a multiple of 10 ms and larger than (1 + connSlaveLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000)
Master_Clock_Accuracy	1	Master clock accuracy. Only valid for a slave	• 0x00: 500 ppm • 0x01: 250 ppm • 0x02: 150 ppm • 0x03: 100 ppm • 0x04: 75 ppm • 0x05: 50 ppm • 0x06: 30 ppm • 0x07: 20 ppm

3.2.2 HCI_LE_ADVERTISING_REPORT_EVENT

Description

The LE advertising report event indicates that one or more Bluetooth devices have responded to an active scan or received some information during a passive scan. The controller may queue these advertising reports and send information from multiple devices in one LE advertising report event. In the current BLE vstack version, only one report is sent per event (num_Reports = 1).

Event parameters

Table 322. HCI_LE_ADVERTISING_REPORT_EVENT event parameters

Parameter	Size	Description	Possible values
Num_Reports	1	Number of responses in this event	0x01
Event_Type[i]	1	Type of advertising report event: 1. ADV_IND: Connectable undirected advertising 2. ADV_DIRECT_IND: Connectable directed advertising 3. ADV_SCAN_IND: Scannable undirected advertising 4. ADV_NONCONN_IND: Non connectable undirected advertising 5. SCAN_RSP: Scan response	• 0x00: ADV_IND • 0x01: ADV_DIRECT_IND • 0x02: ADV_SCAN_IND • 0x03: ADV_NONCONN_IND • 0x04: SCAN_RSP
Address_Type[i]	1	0x00 Public device address; 0x01 Random device address; 0x02 Public identity address (corresponds to resolved private address); 0x03 Random (static) identity address (corresponds to resolved private address)	• 0x00: Public device address • 0x01: Random device address • 0x02: Public identity address • 0x03: Random (static) identity address
Address[i]	6	Public device address or random device address of the device to be connected	-
Length_Data[i]	1	Length of the Data[i] field for each device which responded	0 ... 31
Data[i]	Length_Data[i]	Length_Data[i] octets of advertising or scan response data formatted	-
RSSI[i]	1	N Size: 1 Octet (signed integer) Units: dBm	• 127: RSSI not available • -127 ... 20

3.2.3 HCI_LE_CONNECTION_UPDATE_COMPLETE_EVENT

Description

The LE connection update complete event is used to indicate that the ontroller process to update the connection has completed. On a slave, if no connection parameters are updated, then this event is not issued. On a master, this event is issued if the Connection_Update command was sent.

Event parameters

Table 323. HCI_LE_CONNECTION_UPDATE_COMPLETE_EVENT event parameters

Parameter	Size	Description	Possible values
Status	1	Error code	-
Connection_Handle	2	Connection handle to be used to identify the connection with the peer device.	0x0000 ... 0x0EFF
Conn_Interval	2	Connection interval used on this connection. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE link. It is a multiple of 10 ms and larger than (1 + connSlaveLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)

3.2.4 HCI_LE_READ_REMOTE_FEATURES_COMPLETE_EVENT

Description

The LE read remote features complete event is used to indicate the completion of the process of the controller obtaining the used features of the remote Bluetooth device specified by the Connection_Handle event parameter.

Event parameters

Table 324. HCI_LE_READ_REMOTE_FEATURES_COMPLETE_EVENT event parameters

Parameter	Size	Description	Possible values
Status	1	Error code.	-
Connection_Handle	2	Connection handle to be used to identify the connection with the peer device.	0x0000 ... 0x0EFF
LE_Features	8	Bit mask list of used LE features.	-

3.2.5 HCI_LE_LONG_TERM_KEY_REQUEST_EVENT

Description

The LE long term key request event indicates that the master device is attempting to encrypt or re-encrypt the link and is requesting the long term key from the host.

Event parameters

Table 325. HCI_LE_LONG_TERM_KEY_REQUEST_EVENT event parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle to be used to identify the connection with the peer device.	0x0000 ... 0x0EFF
Random_Number	8	64-bit random number	-
Encrypted_Diversifier	2	16-bit encrypted diversifier	-

3.2.6 HCI_LE_DATA_LENGTH_CHANGE_EVENT

Description

The LE data length change event notifies the host of a change to either the maximum payload length or the maximum transmission time of packets in either direction.

The values reported are the maximum that actually ia used on the connection following the change, except that on the LE coded PHY a packet taking up to 2704 us to transmit may be sent even though the corresponding parameter has a lower value.

Event parameters

Table 326. **HCI_LE_DATA_LENGTH_CHANGE_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle to be used to identify the connection with the peer device.	0x0000 ... 0x0EFF
MaxTxOctets	2	The maximum number of payload octets in a link layer packet that the local controller sends on this connection (connEffectiveMaxTxOctets)	0x001B ... 0x00FB
MaxTxTime	2	The maximum time that the local controller takes to send a link layer packet on this connection (connEffectiveMaxTxTime)	0x0148 ... 0x4290
MaxRxOctets	2	The maximum number of payload octets in a link layer packet that the local controller expects to receive on this connection (connEffectiveMaxRxOctets)	0x001B ... 0x00FB
MaxRxTime	2	The maximum time that the local controller expects to take to receive a link layer packet on this connection (connEffectiveMaxRxTime)	0x0148 ... 0x4290

3.2.7 HCI_LE_READ_LOCAL_P256_PUBLIC_KEY_COMPLETE_EVENT

Description

This event is generated when local P-256 key generation is complete.

Event parameters

Table 327. **HCI_LE_READ_LOCAL_P256_PUBLIC_KEY_COMPLETE_EVENT event parameters**

Parameter	Size	Description	Possible values
Status	1	Error code.	-
Local_P256_Public_Key	64	Local P-256 public key.	-

3.2.8 HCI_LE_GENERATE_DHKEY_COMPLETE_EVENT

Description

This event indicates that LE Diffie Hellman key generation has been completed by the controller.

Event parameters

Table 328. **HCI_LE_GENERATE_DHKEY_COMPLETE_EVENT event parameters**

Parameter	Size	Description	Possible values
Status	1	Error code	-
DHKey	32	Diffie Hellman key	-

3.2.9 HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT

Description

The LE enhanced connection complete event indicates to both of the hosts forming the connection that a new connection has been created. Upon the creation of the connection a Connection_Handle is assigned by the controller, and passed to the host in this event. If the connection establishment fails, this event is provided to the host that had issued the LE_Create_Connection command. If this event is unmasked and LE connection complete event is unmasked, only the LE enhanced connection complete event is sent when a new connection has been completed. This event indicates to the host that issued a LE_Create_Connection command and received a command status event if the connection establishment failed or was successful. The Master_Clock_Accuracy parameter is only valid for a slave. On a master, this parameter is set to 0x00.

Event parameters

Table 329. HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT event parameters

Parameter	Size	Description	Possible values
Status	1	Error code.	-
Connection_Handle	2	Connection handle to be used to identify the connection with the peer device.	0x0000 ... 0x0EFF
Role	1	Role of the local device in the connection.	• 0x00: Master • 0x01: Slave
Peer_Address_Type	1	0x00 Public device address 0x01 Random device address 0x02 Public identity address (corresponds to resolved private address) 0x03 Random (static) identity address (corresponds to resolved private address)	• 0x00: Public device address • 0x01: Random device address • 0x02: Public identity address • 0x03: Random (static) identity address
Peer_Address	6	Public device address, random device address, public identity address or random (static) identity address of the device to be connected.	-
Local_Resolvable_Private_Address	6	Resolvable private address being used by the local device for this connection. This is only valid when the Own_Address_Type is set to 0x02 or 0x03. For other Own_Address_Type values, the controller returns all zeros.	-
Peer_Resolvable_Private_Address	6	Resolvable private address being used by the peer device for this connection. This is only valid for Peer_Address_Type 0x02 and 0x03. For other Peer_Address_Type values, the controller returna all zeros.	-
Conn_Interval	2	Connection interval used on this connection. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE Link. It is a multiple of 10 ms and larger than (1 + connSlaveLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)

Parameter	Size	Description	Possible values
Master_Clock_Accuracy	1	Master clock accuracy. Only valid for a slave.	• 0x00: 500 ppm • 0x01: 250 ppm • 0x02: 150 ppm • 0x03: 100 ppm • 0x04: 75 ppm • 0x05: 50 ppm • 0x06: 30 ppm • 0x07: 20 ppm

3.2.10 HCI_LE_DIRECT_ADVERTISING_REPORT_EVENT

Description

The LE direct advertising report event indicates that directed advertisements have been received where the advertiser is using a resolvable private address for the InitA field in the ADV_DIRECT_IND PDU and the Scanning_Filter_Policy is equal to 0x02 or 0x03, see HCI_LE_Set_Scan_Parameters. Direct_Address_Type and Direct_Addres is the address the directed advertisements are being directed to. Address_Type and address is the address of the advertiser sending the directed advertisements.

Event parameters

Table 330. HCI_LE_DIRECT_ADVERTISING_REPORT_EVENT event parameters

Parameter	Size	Description	Possible values
Num_Reports	1	Number of responses in this event.	0x01
Event_Type[i]	1	Advertising type	0x01: Connectable directed advertising (ADV_DIRECT_IND)
Address_Type[i]	1	0x00 Public device address 0x01 random device address 0x02 public identity address (corresponds to resolved private address) 0x03 random (static) identity address (corresponds to resolved private address)	• 0x00: Public device address • 0x01: Random device address • 0x02: Public identity address • 0x03: Random (static) identity address
Address[i]	6	Public device address, random device address, public identity address or random (static) identity address of the advertising device.	-
Direct_Address_Type[i]	1	0x01 Random device address	0x01: Random device address
Direct_Address[i]	6	Random device address	-
RSSI[i]	1	N Size: 1 Octet (signed integer) Units: dBm	• 127: RSSI not available • -127 ... 20

3.2.11 HCI_LE_PHY_UPDATE_COMPLETE_EVENT

Description

The LE PHY update complete event is used to indicate that the controller has changed the transmitter PHY or receiver PHY in use. If the controller changes the transmitter PHY, the receiver PHY, or both PHYs, this event is issued. If an LE_Set_PHY command was sent and the controller determines that neither PHY changes as a result, it issues this event immediately.

Event parameters

Table 331. HCI_LE_PHY_UPDATE_COMPLETE_EVENT event parameters

Parameter	Size	Description	Possible values
Status	1	Error code.	-
Connection_Handle	2	Connection handle to be used to identify the connection with the peer device.	0x0000 ... 0x0EFF
TX_PHY	1	Transmitter PHY in use	• 0x01: The transmitter PHY for the connection is LE 1M • 0x02: The transmitter PHY for the connection is LE 2M • 0x03: The transmitter PHY for the connection is LE coded (not supported by STM32WB)
RX_PHY	1	Receiver PHY in use	• 0x01: The receiver PHY for the connection is LE 1M • 0x02: The receiver PHY for the connection is LE 2M • 0x03: The receiver PHY for the connection is LE coded (not supported by STM32WB)

3.3 ACI GAP events

In Table 332 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / SO, respectively, for Link layer only / Beacon only / Slave only.

Table 332. ACI GAP events commands list

Command	OpCode	LO	SO	BO
ACI_GAP_LIMITED_DISCOVERABLE_EVENT	0x0400	-	Y	-
ACI_GAP_PAIRING_COMPLETE_EVENT	0x0401	-	Y	-
ACI_GAP_PASS_KEY_REQ_EVENT	0x0402	-	Y	-
ACI_GAP_AUTHORIZATION_REQ_EVENT	0x0403	-	Y	-
ACI_GAP_SLAVE_SECURITY_INITIATED_EVENT	0x0404	-	Y	-
ACI_GAP_BOND_LOST_EVENT	0x0405	-	Y	-
ACI_GAP_PROC_COMPLETE_EVENT	0x0407	-	Y	-
ACI_GAP_ADDR_NOT_RESOLVED_EVENT	0x0408	-	Y	-
ACI_GAP_NUMERIC_COMPARISON_VALUE_EVENT	0x0409	-	Y	-
ACI_GAP_KEYPRESS_NOTIFICATION_EVENT	0x040A	-	Y	-

3.3.1 ACI_GAP_LIMITED_DISCOVERABLE_EVENT

Description

This event is generated by the controller when the limited discoverable mode ends due to timeout. The timeout is 180 seconds.

Event parameters

None

3.3.2 ACI_GAP_PAIRING_COMPLETE_EVENT

Description

This event is generated when the pairing process has completed successfully, or a pairing procedure timeout has occurred, or the pairing has failed. This is to notify the application that there has been a pairing with a remote device so that it can take further actions, or to notify that a timeout has occurred so that the upper layer can decide to disconnect the link.

Event parameters

Table 333. ACI_GAP_PAIRING_COMPLETE_EVENT event parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle on which the pairing procedure is completed	-
Status	1	Pairing status	• 0x00 Success • 0x01 Timeout • 0x02 Failed

Parameter	Size	Description	Possible values
Reason	1	Pairing reason error code (valid in case of pairing failed status)	• 0x00 • 0x01 PASSKEY_ENTRY_FAILED • 0x02: OOB_NOT_AVAILABLE • 0x03: AUTH_REQ_CANNOT_BE_MET • 0x04: CONFIRM_VALUE_FAILED • 0x05: PAIRING_NOT_SUPPORTED • 0x06: INSUFF_ENCRYPTION_KEY_SIZE • 0x07: CMD_NOT_SUPPORTED • 0x08: UNSPECIFIED_REASON • 0x09: VERY_EARLY_NEXT_ATTEMPT • 0x0A: SM_INVALID_PARAMS • 0x0B: SMP_SC_DHKEY_CHECK_FAILED • 0x0C: SMP_SC_NUMCOMPARISON_FAILED

3.3.3 ACI_GAP_PASS_KEY_REQ_EVENT

Description

This event is generated by the security manager to the application when a passkey is required for pairing. When this event is received, the application has to respond with the ACI_GAP_PASS_KEY_RESP command.

Event parameters

Table 334. **ACI_GAP_PASS_KEY_REQ_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the passkey has been requested.	-

3.3.4 ACI_GAP_AUTHORIZATION_REQ_EVENT

Description

This event is generated by the security manager to the application when the application has set that authorization is required for reading/writing of attributes. This event is generated as soon as the pairing is complete. When this event is received, ACI_GAP_AUTHORIZATION_RESP command must be used to respond by the application.

Event parameters

Table 335. **ACI_GAP_AUTHORIZATION_REQ_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which authorization has been requested.	-

3.3.5 ACI_GAP_SLAVE_SECURITY_INITIATED_EVENT

Description

This event is generated when the slave security request is successfully sent to the master.

Event parameters

None

3.3.6 ACI_GAP_BOND_LOST_EVENT

Description

This event is generated when a pairing request is issued in response to a slave security request from a master which has previously bonded with the slave. When this event is received, the upper layer has to issue the command ACI_GAP_ALLOW_REBOND in order to allow the slave to continue the pairing process with the master.

Event parameters

None

3.3.7 ACI_GAP_PROC_COMPLETE_EVENT

Description

This event is sent by the GAP to the upper layers when a procedure previously started has been terminated by the upper layer or has completed for any other reason

Event parameters

Table 336. **ACI_GAP_PROC_COMPLETE_EVENT event parameters**

Parameter	Size	Description	Possible values
Procedure_Code	1	Terminated procedure.	• 0x01: GAP_LIMITED_DISCOVERY_PROC • 0x02: GAP_GENERAL_DISCOVERY_PROC • 0x04: GAP_NAME_DISCOVERY_PROC • 0x08: GAP_AUTO_CONNECTION_ESTABLISHMENT_PROC • 0x10: GAP_GENERAL_CONNECTION_ESTABLISHMENT_PROC • 0x20: GAP_SELECTIVE_CONNECTION_ESTABLISHMENT_PROC • 0x40: GAP_DIRECT_CONNECTION_ESTABLISHMENT_PROC • 0x80: GAP_OBSERVATION_PROC
Status	1	Error code	-
Data_Length	1	Length of data in octets	-
Data	Data_Length	Procedure specific data. For name discovery Procedure: the name of the peer device if the procedure completed successfully.	-

3.3.8 ACI_GAP_ADDR_NOT_RESOLVED_EVENT

Description

This event is sent only by a privacy enabled Peripheral. The event is sent to the upper layers when the peripheral is unsuccessful in resolving the resolvable address of the peer device after connecting to it.

Event parameters

Table 337. **ACI_GAP_ADDR_NOT_RESOLVED_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the private address could not be resolved with any of the stored IRK's.	-

3.3.9 ACI_GAP_NUMERIC_COMPARISON_VALUE_EVENT

Description

This event is sent only during SC v.4.2 Pairing, when numeric comparison association model is selected, in order to show the numeric value generated, and to ask for confirmation to the user. When this event is received, the application has to respond with the ACI_GAP_NUMERIC_COMPARISON_RESP command.

Event parameters

Table 338. **ACI_GAP_NUMERIC_COMPARISON_VALUE_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the underlying pairing	-
Numeric_Value	4	-	-

3.3.10 ACI_GAP_KEYPRESS_NOTIFICATION_EVENT

Description

This event is sent only during SC v.4.2 Pairing, when keypress notifications are supported, in order to show the input type signalled by the peer device, having keyboard only I/O capabilities. When this event is received, no action is required to the user.

Event parameters

Table 339. **ACI_GAP_KEYPRESS_NOTIFICATION_EVEN event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the underlying Pairing	-
Notification_Type	1	Type of keypress input notified/signaled by peer device (having Keyboard only I/O capabilities	-

3.4 ACI GATT/ATT events

In Table 340 "Y" means that the corresponding command applies to the link layer only (LO) (respect to slave only (SO) or beacon only (BO)) variant or slave only of the BLE stack.

Table 340. ACI GATT/ATT events commands list

Command	OpCode	LO	SO	BO
ACI_GATT_ATTRIBUTE_MODIFIED_EVENT	0x0C01	-	Y	-
ACI_GATT_PROC_TIMEOUT_EVENT	0x0C02	-	Y	-
ACI_ATT_EXCHANGE_MTU_RESP_EVENT	0x0C03	-	Y	-
ACI_ATT_FIND_INFO_RESP_EVENT	0x0C04	-	-	-
HCI_LE_LONG_TERM_KEY_REQUEST_EVENT	0x0C05	-	-	-
ACI_ATT_READ_BY_TYPE_RESP_EVENT	0x0C06	-	-	-
ACI_ATT_READ_RESP_EVENT	0x0C07	-	-	-
ACI_ATT_READ_BLOB_RESP_EVENT	0x0C08	-	-	-
ACI_ATT_READ_MULTIPLE_RESP_EVENT	0x0C09	-	-	-
ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT	0x0C0A	-	-	-
ACI_ATT_PREPARE_WRITE_RESP_EVENT	0x0C0C	-	-	-
ACI_ATT_EXEC_WRITE_RESP_EVENT	0x0C0D	-	-	-
ACI_GATT_INDICATION_EVENT	0x0C0E	-	-	-
ACI_GATT_NOTIFICATION_EVENT	0x0C0F	-	-	-
ACI_GATT_PROC_COMPLETE_EVENT	0x0C10	-	Y	-
ACI_GATT_ERROR_RESP_EVENT	0x0C11	-	-	-
ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT	0x0C12	-	-	-
ACI_GATT_WRITE_PERMIT_REQ_EVENT	0x0C13	-	Y	-
ACI_GATT_READ_PERMIT_REQ_EVENT	0x0C14	-	Y	-
ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT	0x0C15	-	Y	-
ACI_GATT_TX_POOL_AVAILABLE_EVENT	0x0C16	-	Y	-
ACI_GATT_SERVER_CONFIRMATION_EVENT	0x0C17	-	Y	-
ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT	0x0C18	-	Y	-
ACI_GATT_READ_EXT_EVENT	0xC1D	-	-	-
ACI_GATT_INDICATION_EXT_EVENT	0xC1E	-	-	-
ACI_GATT_NOTIFICATION_EXT_EVENT	0xC1F	-	-	-

3.4.1 ACI_GATT_ATTRIBUTE_MODIFIED_EVENT

Description

This event is generated to the application by the GATT server when a client modifies any attribute on the server, as consequence of one of the following GATT procedures:

- Write without response
- Signed write without response
- Write characteristic value
- Write long characteristic value - reliable write

Event parameters

Table 341. **Event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	The connection handle which modified the attribute.	-
Attr_Handle	2	Handle of the attribute that was modified.	-
Offset	2	Bits 14-0: Offset from which the write has been performed by the peer device. Bit15 is used as flag: When set to 1 it indicates that more data are to come (fragmented event in case of long attribute data).	-
Attr_Data_Length	2	Length of Attr_Data in octets	-
Attr_Data	Attr_Data_Length	The modified value	-

3.4.2 ACI_GATT_PROC_TIMEOUT_EVENT

Description

This event is generated by the client/server to the application on a GATT timeout (30 seconds). This is a critical event that must not happen during normal operating conditions. It is an indication of either a major disruption in the communication link or a mistake in the application which does not provide a reply to GATT procedures. After this event, the GATT channel is closed and no more GATT communication can be performed. The applications is expected to issue an ACI_GAP_TERMINATE to disconnect from the peer device. It is important to leave an 100 ms blank window before sending the ACI_GAP_TERMINATE, since immediately after this event, system could save important information in non volatile memory.

Event parameters

Table 342. **ACI_GATT_PROC_TIMEOUT_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle on which the GATT procedure has timed out	-

3.4.3 ACI_ATT_EXCHANGE_MTU_RESP_EVENT

Description

This event is generated in response to an Exchange MTU request. See ACI_GATT_EXCHANGE_CONFIG.

Event parameters

Table 343. **ACI_ATT_EXCHANGE_MTU_RESP_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Server_RX_MTU	2	Attribute server receive MTU size	-

3.4.4 ACI_ATT_FIND_INFO_RESP_EVENT

Description

This event is generated in response to a find information request. This event is also generated in response to ACI_GATT_DISC_ALL_CHAR_DESC

Event parameters

Table 344. **ACI_ATT_FIND_INFO_RESP_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Format	1	Format of the hanndle-uuid pairs	-
Event_Data_Length	1	Length of andle_UUID_Pair in octets	-
Handle_UUID_Pair	Event_Data_Length	A sequence of handle-uuid pairs. if format=1, each pair is: [2 octets for handle, 2 octets for UUIDs], if format=2, each pair is: [2 octets for handle, 16 octets for UUIDs]	-

3.4.5 ACI_ATT_FIND_BY_TYPE_VALUE_RESP_EVENT

Description

This event is generated in response to a ACI_ATT_FIND_BY_TYPE_VALUE_REQ.

Event parameters

Table 345. **ACI_ATT_FIND_BY_TYPE_VALUE_RESP_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Num_of_Handle_Pair	1	Number of attribute, group handle pairs	-
Found_Attribute_Handle[i]	2	Found attribute handle	-
Group_End_Handle[i]	2	Group end handle	-

3.4.6 ACI_ATT_READ_BY_TYPE_RESP_EVENT

Description

This event is generated in response to a ACI_ATT_READ_BY_TYPE_REQ. See ACI_GATT_FIND_INCLUDED_SERVICES and ACI_GATT_DISC_ALL_CHAR_DESC.

Event parameters

Table 346. **ACI_ATT_READ_BY_TYPE_RESP_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Handle_Value_Pair_Length	1	The size of each attribute handle-value pair	-
Data_Length	1	Length of Handle_Value_Pair_Data in octets	-
Handle_Value_Pair_Data	Data_Length	Attribute data. A sequence of handle-value pairs: [2 octets for Attribute Handle, (Handle_Value_Pair_Length - 2 octets) for Attribute Value]	-

3.4.7 ACI_ATT_READ_RESP_EVENT

Description

This event is generated in response to a read request. See ACI_GATT_READ_CHAR_VALUE.

Event parameters

Table 347. ACI_ATT_READ_RESP_EVENT event parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Event_Data_Length	1	Length of following data	-
Attribute_Value	Event_Data_Length	The value of the attribute.	-

3.4.8 ACI_ATT_READ_BLOB_RESP_EVENT

Description

This event can be generated during a read long characteristic value procedure. See ACI_GATT_READ_LONG_CHAR_VALUE.

Event parameters

Table 348. ACI_ATT_READ_BLOB_RESP_EVENT event parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	· 0x0000 ... 0x0EFF
Event_Data_Length	1	Length of following data	-
Attribute_Value	Event_Data_Length	Part of the attribute value.	-

3.4.9 ACI_ATT_READ_MULTIPLE_RESP_EVENT

Description

This event is generated in response to a read multiple request.

Event parameters

Table 349. ACI_ATT_READ_MULTIPLE_RESP_EVENT event parameters

Parameter	Size	Description	Possible values
Status	1	Error code	-
Connection_Handle	2	Connection handle for which the command is given.	0x0000 ... 0x0EFF
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Event_Data_Length	1	Length of following data	-
Set_Of_Values	Event_Data_Length	A set of two or more values. A concatenation of attribute values for each of the attribute handles in the request in the order that they were requested.	-

3.4.10 ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT

Description

This event is generated in response to a read by group type request. See ACI_GATT_DISC_ALL_PRIMARY_SERVICES.

Event parameters

Table 350. **ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Attribute_Data_Length	1	The size of each attribute data	-
Data_Length	1	Length of Attribute_Data_List in octets	-
Attribute_Data_List	Data_Length	Attribute data list. A sequence of attribute handle, end group handle, attribute value tuples: [2 octets for Attribute Handle, 2 octets end group handle, (ttribute_Data_Length - 4 octets) for attribute value]	-

3.4.11 ACI_ATT_PREPARE_WRITE_RESP_EVENT

Description

This event is generated in response to a ACI_ATT_PREPARE_WRITE_REQ.

Event parameters

Table 351. **ACI_ATT_PREPARE_WRITE_RESP_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Attribute_Handle	2	The handle of the attribute to be written	-
Offset	2	The offset of the first octet to be written.	-
Part_Attribute_Value_Length	1	Length of Part_Attribute_Value in octets	-
Part_Attribute_Value	Part_Attribute_Value_Length	The value of the attribute to be written	-

3.4.12 ACI_ATT_EXEC_WRITE_RESP_EVENT

Description

This event is generated in response to an Execute Write Request.

Event parameters

Table 352. **ACI_ATT_EXEC_WRITE_RESP_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF

3.4.13 ACI_GATT_INDICATION_EVENT

Description

This event is generated when an indication is received from the server.

Event parameters

Table 353. **ACI_GATT_INDICATION_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Attribute_Handle	2	The handle of the attribute	-
Attribute_Value_Length	1	Length of Attribute_Value in octets	-
Attribute_Value	Attribute_Value_Length	The current value of the attribute	-

3.4.14 ACI_GATT_NOTIFICATION_EVENT

Description

This event is generated when a notification is received from the server.

Event parameters

Table 354. **ACI_GATT_NOTIFICATION_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Attribute_Handle	2	The handle of the attribute	-
Attribute_Value_Length	1	Length of Attribute_Value in octets	-
Attribute_Value	Attribute_Value_Length	The current value of the attribute	-

3.4.15 ACI_GATT_PROC_COMPLETE_EVENT

Description

This event is generated when a GATT client procedure completes either with error or successfully.

Event parameters

Table 355. **ACI_GATT_PROC_COMPLETE_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Error_Code	1	Indicates whether the procedure completed with an error or was successful (see Section 4 Status error codes)	-

3.4.16 ACI_GATT_ERROR_RESP_EVENT

Description

This event is generated when an Error Response is received from the server. The error response can be given by the server at the end of one of the GATT discovery procedures. This does not mean that the procedure ended with an error, but this error event is part of the procedure itself.

Event parameters

Table 356. **ACI_GATT_ERROR_RESP_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Req_Opcode	1	The request that generated this error response	-
Attribute_Handle	2	The attribute handle that generated this error response	-
Error_Code	1	The reason why the request has generated an error response (ATT error codes)	• 0x01: Invalid handle • 0x02: Read not permitted • 0x03: Write not permitted • 0x04: Invalid PDU • 0x05: Insufficient authentication • 0x06: Request not supported • 0x07: Invalid offset • 0x08: Insufficient authorization • 0x09: Prepare queue full • 0x0A: Attribute not found • 0x0B: Attribute not long • 0x0C: Insufficient encryption key size • 0x0D: Invalid attribute value length • 0x0E: Unlikely error • 0x0F: Insufficient encryption • 0x10: Unsupported group type • 0x11: Insufficient resources

3.4.17 ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT

Description

This event can be generated during a "Discover Characteristics By UUID" procedure or a "Read using Characteristic UUID" procedure. The attribute value is a service declaration, when a "Discover characteristics by UUID" has been started. It is the value of the characteristic if a* "Read using characteristic UUID" has been performed.

Event parameters

Table 357. **ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Attribute_Handle	2	The handle of the attribute	-
Attribute_Value_Length	1	Length of Attribute_Value in octets	-
Attribute_Value	Attribute_Value_Length	The attribute value is a service, when a "Discover characteristics by UUID" has been started. It is the value of the characteristic if a "Read using characteristic UUID" has been performed.	-

3.4.18 ACI_GATT_WRITE_PERMIT_REQ_EVENT

Description

This event is given to the application when a write request, write command or signed write command is received by the server from the client. This event is given to the application only if the event bit for this event generation is set when the characteristic was added. When this event is received, the application has to check whether the value being requested for write can be allowed to be written and respond with the command ACI_GATT_WRITE_RESP. The details of the parameters of the command can be found. Based on the response from the application, the attribute value is modified by the stack. If the write is rejected by the application, then the value of the attribute is not modified. In case of a write REQ, an error response is sent to the client, with the error code as specified by the application. In case of write/signed write commands, no response is sent to the client but the attribute is not modified.

Event parameters

Table 358. ACI_GATT_WRITE_PERMIT_REQ_EVENT event parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection on which there was the request to write the attribute	-
Attribute_Handle	2	The handle of the attribute	-
Data_Length	1	Length of data field	-
Data	Data_Length	The data that the client has requested to write	-

3.4.19 ACI_GATT_READ_PERMIT_REQ_EVENT

Description

This event is given to the application when a read request or read blob request is received by the server from the client. This event is given to the application only if the event bit for this event generation is set when the characteristic was added. On receiving this event, the application can update the value of the handle if it desires and when done, it has to send the ACI_GATT_ALLOW_READ command to indicate to the stack that it can send the response to the client.

Event parameters

Table 359. ACI_GATT_READ_PERMIT_REQ_EVENT event parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response.	0x0000 ... 0x0EFF
Attribute_Handle	2	The handle of the attribute	-
Offset	2	Contains the offset from which the read has been requested	-

3.4.20 ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT

Description

This event is given to the application when a read multiple request or read by type request is received by the server from the client. This event is given to the application only if the event bit for this event generation is set when the characteristic was added. On receiving this event, the application can update the values of the handles if it desires and when done, it has to send the ACI_GATT_ALLOW_READ command to indicate to the stack that it can send the response to the client.

Event parameters

Table 360. **ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection which requested to read the attribute	-
Number_of_Handles	1	-	-
Handle[i]	2	-	-

3.4.21 ACI_GATT_TX_POOL_AVAILABLE_EVENT

Description

Each time BLE FW stack raises the error code BLE_STATUS_INSUFFICIENT_RESOURCES (0x64), the ACI_GATT_TX_POOL_AVAILABLE_EVENT event is generated as soon as there are at least two buffers available for notifications or write commands.

Event parameters

Table 361. **ACI_GATT_TX_POOL_AVAILABLE_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the request	-
Available_Buffers	2	Number of buffers available	-

3.4.22 ACI_GATT_SERVER_CONFIRMATION_EVENT

Description

This event is generated when the client has sent the confirmation to a previously sent indication.

Event parameters

Table 362. **ACI_GATT_SERVER_CONFIRMATION_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the event	-

3.4.23 ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT

Description

This event is given to the application when a prepare write request is received by the server from the client. This event is given to the application only if the event bit for this event generation is set when the characteristic was added. When this event is received, the application has to check whether the value being requested for write can be allowed to be written and respond with the command ACI_GATT_WRITE_RESP. Based on the response from the application, the attribute value is modified by the stack. If the write is rejected by the application, then the value of the attribute is not modified and an error response is sent to the client, with the error code as specified by the application.

Event parameters

Table 363. **ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection on which there was the request to write the attribute	-
Attribute_Handle	2	The handle of the attribute	-
Offset	2	The offset from which the prepare write has been requested	-
Data_Length	1	Length of Data field	-
Data	Data_Length	The data that the client has requested to write	-

3.4.24 ACI_GATT_READ_EXT_EVENT

Description

When it is enabled with ACI_GATT_SET_EVENT_MASK, this event is generated instead of ACI_ATT_READ_RESP_EVENT / ACI_ATT_READ_BLOB_RESP_EVENT / ACI_ATT_READ_MULTIPLE_RESP_EVENT. This event should be used instead of those events when ATT_MTU > (BLE_EVT_MAX_PARAM_LEN - 4), i.e. ATT_MTU > 251 for BLE_EVT_MAX_PARAM_LEN default value.

Event parameters

Table 364. **ACI_GATT_READ_EXT_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to response	0x0000 ... 0x0EFF
Offset	2	Bits 14-0: offset in octets from which Attribute_Value data starts. Bit 15 is used as flag: when set to 1 it indicates that more data are to come (fragmented event in case of long attribute data)	-
Event_Data_Length	2	Length of following Data	-
Attribute_Value	Event_Data_Length	The value of the attribute(s)	-

3.4.25 ACI_GATT_INDICATION_EXT_EVENT

Description

When it is enabled with ACI_GATT_SET_EVENT_MASK and when an indication is received from the server, this event is generated instead of ACI_GATT_INDICATION_EVENT. This event is used instead of ACI_GATT_INDICATION_EVENT when ATT_MTU > (BLE_EVT_MAX_PARAM_LEN - 4) i.e. ATT_MTU > 251 for BLE_EVT_MAX_PARAM_LEN default value.

Event parameters

Table 365. **ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response	0x0000 ... 0x0EFF
Attribute_Handle	2	The handle of the attribute	-
Offset	2	Bits 14-0: offset in octets from which Attribute_Value data starts. Bit 15 is used as flag: when set to 1 it indicates that more data are to come (fragmented event in case of long attribute data).	-
Data_Length	2	Length of Attribute_Value in octets	-
Data	Attribute_Value_Length	The current value of the attribute	-

3.4.26 ACI_GATT_NOTIFICATION_EXT_EVENT

Description

When it is enabled with ACI_GATT_SET_EVENT_MASK and when a notification is received from the server, this event is generated instead of ACI_GATT_NOTIFICATION_EVENT. This event is used instead of ACI_GATT_NOTIFICATION_EVENT when ATT_MTU > (BLE_EVT_MAX_PARAM_LEN - 4) i.e. ATT_MTU > 251 for BLE_EVT_MAX_PARAM_LEN default value.

Event parameters

Table 366. **ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle related to the response	0x0000 ... 0x0EFF
Attribute_Handle	2	The handle of the attribute	-
Offset	2	Bits 14-0: offset in octets from which Attribute_Value data starts. Bit 15 is used as flag: when set to 1 it indicates that more data are to come (fragmented event in case of long attribute data).	-
Data_Length		Length of Attribute_Value in octets	-
Data	Attribute_Value_Length	The current value of the attribute	-

3.5 ACI L2CAP events

In Table 367 "Y" means that the corresponding command applies to the link layer only (LO) (respect to slave only (SO) or beacon only (BO)) variant or slave only of the BLE stack.

Table 367. ACI L2CAP events commands list

Command	OpCode	LO	SO	BO
ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT	0x0800	-	Y	-
ACI_L2CAP_PROC_TIMEOUT_EVENT	0x0801	-	Y	-
ACI_L2CAP_CONNECTION_UPDATE_REQ_EVENT	0x0802	-	-	-
ACI_L2CAP_COMMAND_REJECT_EVENT	0x080A	-	Y	-

3.5.1 ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT

Description

This event is generated when the master responds to the connection update request packet with a connection update response packet.

Event parameters

Table 368. ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT event parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle referring to the COS channel where the disconnection has been received	-
Result	3	-	-

3.5.2 ACI_L2CAP_PROC_TIMEOUT_EVENT

Description

This event is generated when the master does not respond to the connection update request packet with a connection update response packet or a command reject packet within 30 seconds.

Event parameters

Table 369. ACI_L2CAP_PROC_TIMEOUT_EVENT event parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection related to this L2CAP procedure.	-
Data_Length	1	Length of following data	-
Data	Data_Length	-	-

3.5.3 ACI_L2CAP_CONNECTION_UPDATE_REQ_EVENT

Description

The event is given by the L2CAP layer when a connection update request is received from the slave. The upper layer which receives this event has to respond by sending a ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_RESP command.

Event parameters

Table 370. **ACI_L2CAP_CONNECTION_UPDATE_REQ_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection related to this L2CAP procedure.	-
Identifier	1	This is the identifier which associate the request to the response.	-
L2CAP_Length	2	Length of the L2CAP connection update request.	-
Interval_Min	2	Minimum value for the connection event interval. This is less than or equal to Conn_Interval_Max. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Interval_Max	2	Maximum value for the connection event interval. This is greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.50 ms) ... 0x0C80 (4000.00 ms)
Slave_Latency	2	Slave latency for the connection in number of connection events.	0x0000 ... 0x01F3
Timeout_Multiplier	2	Defines connection timeout parameter in the following manner: Timeout Multiplier * 10 ms.	-

3.5.4 ACI_L2CAP_COMMAND_REJECT_EVENT

Description

This event is generated when the master responds to the connection update request packet with a command reject packet.

Event parameters

Table 371. **ACI_L2CAP_COMMAND_REJECT_EVENT event parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle referring to the COS channel where the disconnection has been received.	-
Identifier	1	This is the identifier which associate the request to the response.	-
Reason	2	Reason	-
Data_Length	1	Length of following data	-
Data	Data_Length	Data field associated with reason	-

3.6 ACI HAL events

In Table 372 "Y" means that the corresponding command applies to the link layer only (LO) (respect to slave only (SO) or beacon only (BO)) variant or slave only of the BLE stack.

Table 372. ACI HAL events commands list

Command	OpCode	LO	SO	BO
ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT	0x0004	Y	Y	Y
ACI_HAL_SCAN_REQ_REPORT_EVENT	0x0005	-	-	-
ACI_HAL_FW_ERROR_EVENT	0x0006	-	Y	-

3.6.1 ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT

Description

This event is generated when the device completes a radio activity and provide information when a new radio activity is performed. The provided information includes type of radio activity and absolute time in system ticks when a new radio activity is schedule, if any. The application uses this information to schedule user activities synchronous to selected radio activities. A command ACI_HAL_SET_RADIO_ACTIVITY_MASK is provided to enable radio activity events of user interests, by default no events are enabled. The enabling radio events in application with intense radio activity could lead to a fairly high rate of events generated. The application use cases includes synchronizing notification with connection interval, switching antenna at the end of advertising or performing flash erase operation while radio is idle.

Event parameters

Table 373. ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT event parameters

Parameter	Size	Description	Possible values
Last_State	1	Completed radio events	• 0x00: Idle • 0x01: Advertising • 0x02: Connection event slave • 0x03: Scanning • 0x04: Connection request • 0x05: Connection event slave • 0x06: TX test mode • 0x07: RX test mode
Next_State	1	Incoming radio events	• 0x00: Idle • 0x01: Advertising • 0x02: Connection event slave • 0x03: Scanning • 0x04: Connection request • 0x05: Connection event slave • 0x06: TX test mode • 0x07: RX test mode
Next_State_SysTime	4	32 bit absolute current time expressed in internal time units.	-

3.6.2 ACI_HAL_SCAN_REQ_REPORT_EVENT

Description

This event is reported to the application after a scan request is received and a scan reponse is scheduled to be transmitted.

Event parameters

Table 374. **ACI_HAL_SCAN_REQ_REPORT_EVENT event parameters**

Parameter	Size	Description	Possible values
RSSI	1	N Size: 1 Octet (signed integer) Units: dBm	• 127: RSSI not available • -127 ... 20
Peer_Address_Type	1	0x00 Public device address 0x01 random device address 0x02 public identity address (corresponds to resolved private address) 0x03 random (static) identity address (corresponds to resolved private address)	• 0x00: Public device address • 0x01: Random device address • 0x02: Public identity address • 0x03: Random (static) identity address
Peer_Address	6	Public device address or random device Address of the peer device	-

3.6.3 ACI_HAL_FW_ERROR_EVENT

Description

This event is generated to report firmware error informations.

Event parameters

Table 375. **ACI_HAL_FW_ERROR_EVENT event parameters**

Parameter	Size	Description	Possible values
FW_Error_Type	1	FW error type	• 0x01: L2CAP recombination failure • 0x02: Gatt unexpected peer message • 0x03: NVM level warning
Data_Length	1	Length of data in octets	-
Data	Data_Length	The error event info	-

4 Status error codes

Status error codes are used for the return status of all commands. Only the codes from 0 to 0x3E are used for HCI commands (see Core Specification v5.2, Vol. 2, part D), while more codes are defined for ACI commands (see table below).

Table 376. Status error codes description

Status error code	Description
0x00	Success
0x01	Unknown HCI command
0x02	Unknown connection identifier
0x03	Hardware failure
0x05	Authentication failure
0x06	PIN or key missing
0x07	Memory capacity exceeded
0x08	Connection timeout
0x09	Connection limit exceeded
0x0B	ACL connection already exists
0x0C	Command disallowed
0x0D	Connection rejected due to limited resources
0x0E	Connection rejected due to security reasons
0x0F	Connection rejected due to unacceptable BD_ADDR
0x10	Connection accept timeout exceeded
0x11	Unsupported feature or parameter value
0x12	Invalid HCI command parameters
0x13	Remote user terminated connection
0x14	Remote device terminated connection due to low Resources
0x15	Remote device terminated connection due to power-off
0x16	Connection terminated by local host
0x17	Repeated attempts
0x18	Pairing not allowed
0x19	Unknown LMP PDU
0x1A	Unsupported remote feature / unsupported LMP feature
0x1E	Invalid LMP parameters
0x1F	Unspecified error
0x20	Unsupported LMP parameter value
0x21	Role change not allowed
0x22	LMP response timeout / LL response timeout
0x23	LMP error transaction collision
0x24	LMP PDU not allowed
0x25	Encryption mode not acceptable
0x26	Link key cannot be changed

Status error code	Description
0x28	Instant passed
0x29	Pairing with unit key not supported
0x2A	Different transaction collision
0x2E	Channel assessment not supported
0x2F	Insufficient security
0x30	Parameter out of mandatory range
0x32	Role switch pending
0x34	Reserved slot violation
0x35	Role switch failed
0x37	Secure simple pairing not supported by host
0x38	Host busy - pairing
0x39	Connection rejected due to no suitable channel fFound
0x3A	Controller busy
0x3B	Unacceptable connection interval
0x3C	Directed advertising timeout
0x3D	Connection terminated due to MIC failure
0x3E	Connection failed to be established
0x40	Unknown connection identifier at SMP level
0x41	Failed
0x42	Invalid parameters
0x43	Busy
0x45	Pending
0x46	Not allowed
0x47	Host error
0x48	Out of memory
0x50	Invalid CID
0x59	Device in blacklist
0x5A	CSRK not found
0x5B	IRK not found
0x5C	Device not found in DB
0x5D	Security DB full
0x5E	Device not bonded
0x5F	Insufficient encryption key size
0x60	Invalid handle
0x61	Out of handles
0x62	Invalid operation
0x63	Characteristic already exist
0x64	Insufficient resources
0x65	Security permission error
0x70	Address not resolved
0x82	No valid slot

Status error code	Description
0x83	Short window
0x84	New interval failed
0x85	Too large interval
0x86	Slot length failed

Revision history

Table 377. **Document revision history**

Date	Version	Changes
20-Feb-2019	1	Initial release.
04-Jun-2019	2	Updated Table 6. HCI_SET_EVENT_MASK input parameters, Table 315. HCI_HARDWARE_ERROR_EVENT event parameters, Table 375. ACI_HAL_FW_ERROR_EVENT event parameters
17-Jun-2019	3	Updated: • Table 226. ACI_GATT_UPDATE_CHAR_VALUE input parameters, Table 234. ACI_GATT_SET_EVENT_MASK input parameters, Table 262. ACI_GATT_READ_CHAR_VALUE input parameters, Table 266. ACI_GATT_READ_LONG_CHAR_VALUE input parameters, Table 270. ACI_GATT_WRITE_CHAR_VALUE input parameters, Table 272. ACI_GATT_WRITE_LONG_CHAR_VALUE input parameters, Table 284. ACI_GATT_WRITE_WITHOUT_RESP input parameters, Table 286. ACI_GATT_SIGNED_WRITE_WITHOUT_RESP input parameters, Table 300. ACI_GATT_UPDATE_CHAR_VALUE_EXT input parameters, Table 340. ACI GATT/ATT events commands list Added: • Section 3.4.25 ACI_GATT_INDICATION_EXT_EVENT and Section 3.4.26 ACI_GATT_NOTIFICATION_EXT_EVENT
25-Sep-2019	4	Updated: • Section 2.4.18 ACI_GAP_CONFIGURE_WHITELIST, Section 2.5.14 ACI_ATT_READ_BY_TYPE_REQ, Section 2.5.25 ACI_GATT_READ_USING_CHAR_UUID, Section 3.4.25 ACI_GATT_INDICATION_EXT_EVENT, Section 4 Status error codes • Table 189. ACI_GAP_TERMINATE_GAP_PROC input parameters, Table 234. ACI_GATT_SET_EVENT_MASK input parameters, Table 312. HCI_DISCONNECTION_COMPLETE_EVENT event parameters, Table 336. ACI_GAP_PROC_COMPLETE_EVENT event parameters, Table 340. ACI GATT/ATT events commands list, Table 341. Event parameters, Table 365. ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT event parameters, Table 366. ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT event parameters, Table 376. Status error codes description Added Section 3.4.24 ACI_GATT_READ_EXT_EVENT Removed section 4.2 ATT error codes
07-Jan-2020	5	Updated: • Table 109. ACI_HAL_WRITE_CONFIG_DATA input parameters, Table 111. ACI_HAL_READ_CONFIG_DATA input parameters, Table 333. ACI_GAP_PAIRING_COMPLETE_EVENT event parameters • Section 2.5.35 ACI_GATT_WRITE_WITHOUT_RESP, Section 2.5.36 ACI_GATT_SIGNED_WRITE_WITHOUT_RESP

Date	Version	Changes
26-May-2020	6	Updated: • Table 6. HCI_SET_EVENT_MASK input parameters, Table 109. ACI_HAL_WRITE_CONFIG_DATA input parameters, Table 111. ACI_HAL_READ_CONFIG_DATA input parameters, Table 312. HCI_DISCONNECTION_COMPLETE_EVENT event parameters, Table 333. ACI_GAP_PAIRING_COMPLETE_EVENT event parameters, Table 355. ACI_GATT_PROC_COMPLETE_EVENT event parameters, Table 375. ACI_HAL_FW_ERROR_EVENT event parameters, Table 376. Status error codes description • Title of Section 2.1.34 HCI_LE_READ_REMOTE_FEATURES and Section 3.2.4 HCI_LE_READ_REMOTE_FEATURES_COMPLETE_EVENT • Section 2.4.22 ACI_GAP_START_LIMITED_DISCOVERY_PROC, Section 2.4.23 ACI_GAP_START_GENERAL_DISCOVERY_PROC Added note in Section 2.4.10 ACI_GAP_INIT Removed section "HCI_DATA_BUFFER_OVERFLOW_EVENT"
16-Jul-2020	7	Added three columns in Table 1. HCI commands list, Table 97. HCI testing commands list, Table 107. HAL commands list, Table 135. GAP commands list, Table 216. GATT/ATT commands list, Table 306. L2CAP commands list, Table 311. HCI events commands list, Table 320. HCI LE META events commands list, Table 332. ACI GAP events commands list, Table 340. ACI GATT/ATT events commands list, Table 367. ACI L2CAP events commands list and Table 372. ACI HAL events commands list. Updated Section 3.2.2 HCI_LE_ADVERTISING_REPORT_EVENT. Removed section 3.6.4 ACI_HAL_DATAPUMP_SENT_EVENT.
03-Nov-2020	8	Updated document title, Introduction, Section 1 General information, Section 2.1 HCI commands, Section 2.2 HCI testing commands, Section 2.3 HAL commands, Section 2.4 GAP commands, Section 2.5 GATT/ATT commands, Section 2.6 L2CAP commands and Section 3.1 HCI events. Updated Table 1. HCI commands list, Table 107. HAL commands list and Table 333. ACI_GAP_PAIRING_COMPLETE_EVENT event parameters. Removed former section 2.3.12 ACI_HAL_SET_SMP_ENG_CONFIG. Minor text edits across the whole document.
21-Jan-2021	9	Updated Introduction, Section 3.2 HCI LE META events, Section 3.3 ACI GAP events and Section 4 Status error codes. Updated Table 119. ACI_HAL_GET_LINK_STATUS output parameters. Minor text edits across the whole document.

Contents

List of tables