life.augmented

AN5270

Application note

Introduction to STM32WB Bluetooth® Low Energy wireless interface

Introduction

Bluetooth® Low Energy 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.

Bluetooth® Low Energy considerably reduces power consumption and cost compared to the standard Bluetooth, while maintaining a similar communication range.

Standard HCI commands are defined in the Bluetooth specification core V6.0, of which the Bluetooth® Low Energy specification is a part.

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 Bluetooth® Low Energy (BLE) stack variant, namely LO / BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 1. HCI commands list

Command	OpCode	LO	PO	BO	BF	LB
HCI_DISCONNECT	0x0406	Y	-	-	-	Y
HCI_READ_REMOTE_VERSION_INFORMATION	0x041D	Y	Y	-	Y	Y
HCI_SET_EVENT_MASK	0x0C01	Y	Y	Y	Y	Y
HCI_RESET	0x0C03	Y	Y	Y	Y	Y
HCI_READ_TRANSMIT_POWER_LEVEL	0x0C2D	Y	Y	-	Y	Y
HCI_SET_CONTROLLER_TO_HOST_FLOW_CONTROL	0x0C31	Y	-	-	-	Y
HCI_HOST_BUFFER_SIZE	0x0C33	Y	-	-	-	Y
HCI_HOST_NUMBER_OF_COMPLETED_PACKETS	0x0C35	Y	-	-	-	Y
HCI_READ_LOCAL_VERSION_INFORMATION	0x1001	Y	Y	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	Y	Y
HCI_READ_RSSI	0x1405	Y	Y	-	Y	Y
HCI_LE_SET_EVENT_MASK	0x2001	Y	Y	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	-	Y
HCI_LE_SET_ADVERTISING_PARAMETERS	0x2006	Y	-	Y	-	Y
HCI_LE_READ_ADVERTISING_PHYSICAL_CHANNEL_TX_POWER	0x2007	Y	-	Y	-	Y
HCI_LE_SET_ADVERTISING_DATA	0x2008	Y	Y	Y	Y	Y
HCI_LE_SET_SCAN_RESPONSE_DATA	0x2009	Y	Y	Y	Y	Y
HCI_LE_SET_ADVERTISING_ENABLE	0x200A	Y	-	Y	-	Y
HCI_LE_SET_SCAN_PARAMETERS	0x200B	Y	-	Y	-	Y
HCI_LE_SET_SCAN_ENABLE	0x200C	Y	-	Y	-	Y
HCI_LE_CREATE_CONNECTION	0x200D	Y	-	-	-	Y
HCI_LE_CREATE_CONNECTION_CANCEL	0x200E	Y	-	-	-	Y
HCI_LE_READ_FILTER_ACCEPT_LIST_SIZE	0x200F	Y	-	Y	-	Y
HCI_LE_CLEAR_FILTER_ACCEPT_LIST	0x2010	Y	-	Y	-	Y
HCI_LE_ADD_DEVICE_TO_FILTER_ACCEPT_LIST	0x2011	Y	-	Y	-	Y
HCI_LE_REMOVE_DEVICE_FROM_FILTER_ACCEPT_LIST	0x2012	Y	-	Y	-	Y
HCI_LE_CONNECTION_UPDATE	0x2013	Y	-	-	-	Y
HCI_LE_SET_HOST_CHANNEL_CLASSIFICATION	0x2014	Y	-	-	Y	Y
HCI_LE_READ_CHANNEL_MAP	0x2015	Y	Y	-	Y	Y
HCI_LE_READ_REMOTE_FEATURES	0x2016	Y	Y	-	Y	Y

Command	OpCode	LO	PO	BO	BF	LB
HCI_LE_ENCRYPT	0x2017	Y	Y	-	Y	Y
HCI_LE_RAND	0x2018	Y	Y	Y	Y	Y
HCI_LE_ENABLE_ENCRYPTION	0x2019	Y	-	-	-	Y
HCI_LE_LONG_TERM_KEY_REQUEST_REPLY	0x201A	Y	Y	-	Y	Y
HCI_LE_LONG_TERM_KEY_REQUEST_NEGATIVE_REPLY	0x201B	Y	Y	-	Y	Y
HCI_LE_READ_SUPPORTED_STATES	0x201C	Y	-	Y	-	Y
HCI_LE_SET_DATA_LENGTH	0x2022	Y	Y	-	Y	Y
HCI_LE_READ_SUGGESTED_DEFAULT_DATA_LENGTH	0x2023	Y	Y	-	Y	Y
HCI_LE_WRITE_SUGGESTED_DEFAULT_DATA_LENGTH	0x2024	Y	Y	-	Y	Y
HCI_LE_READ_LOCAL_P256_PUBLIC_KEY	0x2025	Y	Y	-	Y	Y
HCI_LE_GENERATE_DHKEY	0x2026	Y	-	-	-	Y
HCI_LE_ADD_DEVICE_TO_RESOLVING_LIST	0x2027	Y	-	-	-	Y
HCI_LE_REMOVE_DEVICE_FROM_RESOLVING_LIST	0x2028	Y	-	-	-	Y
HCI_LE_CLEAR_RESOLVING_LIST	0x2029	Y	-	-	-	Y
HCI_LE_READ_RESOLVING_LIST_SIZE	0x202A	Y	-	-	-	Y
HCI_LE_READ_PEER_RESOLVABLE_ADDRESS	0x202B	Y	Y	-	Y	Y
HCI_LE_READ_LOCAL_RESOLVABLE_ADDRESS	0x202C	Y	Y	-	Y	Y
HCI_LE_SET_ADDRESS_RESOLUTION_ENABLE	0x202D	Y	-	-	-	Y
HCI_LE_SET_RESOLVABLE_PRIVATE_ADDRESS_TIMEOUT	0x202E	Y	Y	-	Y	Y
HCI_LE_READ_MAXIMUM_DATA_LENGTH	0x202F	Y	Y	-	Y	Y
HCI_LE_READ_PHY	0x2030	Y	-	-	Y	Y
HCI_LE_SET_DEFAULT_PHY	0x2031	Y	-	-	Y	Y
HCI_LE_SET_PHY	0x2032	Y	Y	-	Y	Y
HCI_LE_SET_ADVERTISING_SET_RANDOM_ADDRESS	0x2035	Y	-	-	-	Y
HCI_LE_SET_EXTENDED_ADVERTISING_PARAMETERS	0x2036	Y	-	-	-	-
HCI_LE_SET_EXTENDED_ADVERTISING_DATA	0x2037	Y	-	-	-	-
HCI_LE_SET_EXTENDED_SCAN_RESPONSE_DATA	0x2038	Y	-	-	-	-
HCI_LE_SET_EXTENDED_ADVERTISING_ENABLE	0x2039	Y	-	-	-	-
HCI_LE_READ_MAXIMUM_ADVERTISING_DATA_LENGTH	0x203A	Y	-	-	-	-
HCI_LE_READ_NUMBER_OF_SUPPORTED_ADVERTISING_SETS	0x203B	Y	-	-	-	-
HCI_LE_REMOVE_ADVERTISING_SET	0x203C	Y	-	-	-	-
HCI_LE_CLEAR_ADVERTISING_SETS	0x203D	Y	-	-	-	-
HCI_LE_SET_EXTENDED_SCAN_PARAMETERS	0x2041	Y	-	-	-	-
HCI_LE_SET_EXTENDED_SCAN_ENABLE	0x2042	Y	-	-	-	-
HCI_LE_EXTENDED_CREATE_CONNECTION	0x2043	Y	-	-	-	-
HCI_LE_READ_TRANSMIT_POWER	0x204B	Y	-	Y	-	Y
HCI_LE_READ_RF_PATH_COMPENSATION	0x204C	Y	-	-	-	-
HCI_LE_WRITE_RF_PATH_COMPENSATION	0x204D	Y	-	-	-	-
HCI_LE_SET_PRIVACY_MODE	0x204E	Y	Y	-	Y	Y
HCI_LE_GENERATE_DHKEY_V2	0x205E	Y	-	-	-	Y

2.1.1 HCI_DISCONNECT

This command is used to terminate an existing connection. The Connection_Handle parameter indicates the connection to be disconnected, and the Reason parameter indicates the reason for ending it. The remote controller receives the Reason parameter in the HCI_DISCONNECTION_COMPLETE_EVENT. All synchronous connections on a physical link must be disconnected before the ACL connection on the same physical connection is disconnected. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.1.6].

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

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

This command obtains the version information values for the remote device identified by the Connection_Handle parameter, which must be a Connection_Handle for an ACL or LE connection. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.1.23].

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

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

This command is used to control the events generated by the HCI for the host. If the bit in the Event_Mask is set to 1 the associated event is enabled. For an LE controller, the LE Meta event bit in the Event_Mask enables or disables all LE events in the event. The host must deal with each occurring event, the event mask allows it to control if it is interrupted. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.3.1].

Table 6. HCI_SET_EVENT_MASK input parameters

Parameter	Size	Description	Possible values
Event_Mask	8	Event mask. Default: 0x2000FFFFFFFFFFFF	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

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

This command resets the link layer on an LE controller, but does not affect the used HCI transport layer, as these layers can have their own reset mechanisms. After the reset is completed, the current operational state is lost, the controller enters standby mode, and automatically reverts to the default for the parameters for which default values are defined in the specification. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.3.2].

Note: This command does not perform a hardware reset. The host does not send additional HCI commands before it receives the command Complete event related to the Reset command.

Input parameters: none

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

This command reads the values for the Transmit_Power_Level parameter for the Connection_Handle specified for an ACL connection. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.3.35].

Table 9. HCI_READ_TRANSMIT_POWER_LEVEL input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the Connection_Handle 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

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), in dBm	-30 … 20

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.6 HCI_SET_CONTROLLER_TO_HOST_FLOW_CONTROL

This command is used by the host to turn flow control on or off for data and/or voice sent from the controller.

If the control is turned off, the host does not send the HCI_HOST_NUMBER_OF_COMPLETED_PACKETS command: this command is ignored by the controller if it is sent when flow control is off.

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

If the control is turned on for HCI ACL and HCI synchronous data packets, the host sends HCI_HOST_NUMBER_OF_COMPLETED_PACKETS commands for both ACL and synchronous connections. If no connection exists, only the Flow_Control_Enable parameter is changed.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.3.38].

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 from controller to host (default). • 0x01: Flow control on for HCI ACL data packets and off for HCI synchronous. Data packets from controller to host. • 0x02: Flow control off for HCI ACL data packets and on for HCI synchronous. Data packets from controller to host. • 0x03: Flow control on both for HCI ACL data packets and HCI synchronous. Data packets from controller to host.

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_COMPLETE_EVENT

2.1.7 HCI_HOST_BUFFER_SIZE

This 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. The controller segments the data to transmit according to their size, so that the HCI data packets contain data up to these sizes.

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

The HCI_SET_CONTROLLER_TO_HOST_FLOW_CONTROL command is used to turn flow control on or off.

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

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

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.3.39].

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

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 than or equal to 251 bytes.	251 … 65535
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 host data buffers.	1 … 65535
Host_Total_Num_Synchronous_Data_Packets	2	Total number of HCI synchronous data packets that can be stored in the host data buffers.	-

Table 14. HCI_HOST_BUFFER_SIZE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.8 HCI_HOST_NUMBER_OF_COMPLETED_PACKETS

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

Based on this information, the Host_Total_Num_ACL_Data_Packets and Host_Total_Num_Synchronous_Data_Packets parameters of the HCI_HOST_BUFFER_SIZE command, the controller determines for which Connection_Handles the next HCI data packets must be sent. The command is issued only if flow control 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. When the host has completed one or more HCI data packet(s), it sends a HCI_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 HCI_SET_CONTROLLER_TO_HOST_FLOW_CONTROL command is used to turn flow control on or off. If the control is turned on, the HCI_HOST_BUFFER_SIZE command is always sent by the host after a power-on or a reset before the first HCI_HOST_NUMBER_OF_COMPLETED_PACKETS command is sent.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.3.40].

Note: This command is special: no event is generated after the command has completed. The command can be sent at any time by the host when there is at least one connection, or if the controller is in local loop back mode, independently from other commands. The normal flow control is not used.

Table 15. **HCI_HOST_NUMBER_OF_COMPLETED_PACKETS input parameters**

Parameter	Size	Description	Possible values
Number_Of_Handles	1	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	Number of HCI data packets completed for the associated Connection_Handle since the previous time the event was returned.	0x0000-0xFFFF

Output parameters: none

Unless masked away, no events are generated after this command has completed. However, if the command contains one or more invalid parameters, the controller returns a HCI_COMMAND_COMPLETE_EVENT with a failure status indicating the invalid HCI command parameters error code. The host may send the command at any time when there is at least one connection, or if the controller is in local loop back mode. The normal flow control is not used for this command.

2.1.9 HCI_READ_LOCAL_VERSION_INFORMATION

This command reads the values of the version information for the local controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.4.1].

Input parameters: none

Table 16. **HCI_READ_LOCAL_VERSION_INFORMATION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
HCI_Version	1	Version of the HCI specification supported by the controller. See Bluetooth assigned numbers.	-
HCI_Subversion	2	Revision of the HCI implementation in the controller (vendor specific)	-
LMP_Version	1	Version of the current LMP supported by the controller. See Bluetooth assigned numbers.	-
Company_Identifier	2	Company identifier for the manufacturer of the controller. See Bluetooth assigned numbers.	-
LMP_Subversion	2	Subversion of the current LMP in the controller (implementation dependent)	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.10 HCI_READ_LOCAL_SUPPORTED_COMMANDS

This command reads the list of HCI commands supported for the local controller, and returns the Supported_Commands configuration parameter. If a command is supported, the feature underlying that command is supported. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.4.2].

Input parameters: none

Table 17. **HCI_READ_LOCAL_SUPPORTED_COMMANDS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Parameter	Size	Description	Possible values
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 it. Unsupported or undefined commands are set to 0.	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.11 HCI_READ_LOCAL_SUPPORTED_FEATURES

This command requests a list of the supported features for the local controller, and returns a list of the LMP features. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.4.3].

Input parameters: none

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

On an LE controller, this command reads the public device address. If the controller does not have one, the value 0x000000000000 is returned. The public address is the same as the BD_ADDR. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.4.6].

Input parameters: none

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

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, with ±6 dB accuracy. If the RSSI cannot be read, the RSSI metric is set to 127. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.5.4].

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

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

This 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, the associated event is enabled. The host must deal with each event generated by an LE controller. The event mask allows the host to control which events interrupt it. To generate an LE event, the LE Meta-Event bit in the Event_Mask must be set. If not, LE events are not generated, regardless of how the LE_Event_Mask is set. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.1].

Table 22. HCI_LE_SET_EVENT_MASK input parameters

Parameter	Size	Description	Possible values
LE_Event_Mask	8	LE event mask. Default: 0x000000000003185F	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

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

This 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 transmitted data according to these values, so that the HCI data packets contain data up to this size. The command also returns the total number of HCI LE ACL data packets stored in the data buffers of the controller. The command must be issued by the host before it sends any data to an LE controller. If the controller returns a length value of 0, the host uses the command to determine the size of the data buffers.

Note: Both the HCI_READ_BUFFER_SIZE and HCI_LE_READ_BUFFER_SIZE commands can return non-0 values for the buffer length and number of packets parameters.

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 stored in the data buffers of the controller. The host determines how the buffers must 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.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.2]

Input parameters: none

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 stored in the data buffers of the controller	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.16 HCI_LE_READ_LOCAL_SUPPORTED_FEATURES

This command requests the list of the supported LE features for the controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.3].

Input parameters: none

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

This command is used by the host to set the LE random device address in the controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.4].

Table 26. HCI_LE_SET_RANDOM_ADDRESS input parameters

Parameter	Size	Description	Possible values
Random_Address	6	Random device address	-

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

This command is used by the host to set the advertising parameters.

Advertising_Interval_Min is lower than or equal to the Advertising_Interval_Max. These parameters cannot have the same value, to let the controller determine the best interval, considering other activities. For high duty cycle directed advertising (when Advertising_Type is 0x01) these parameters are ignored.

Advertising_Type determines the packet type used for advertising when enabled. Advertising_Interval_Min and Advertising_Interval_Max are set at least to 0x00A0 (100 ms) if Advertising_Type is set to 0x02 (ADV_SCAN_IND) or 0x03 (ADV_NONCONN_IND).

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 HCI_LE_SET_RANDOM_ADDRESS command. If directed advertising is performed (Advertising_Type set to 0x01 or 0x04), the Direct_Address_Type and Direct_Address are valid, otherwise they are ignored by the controller.

Advertising_Channel_Map indicates the channels used when transmitting advertising packets. At least one channel bit is set. The Advertising_Filter_Policy parameter is ignored when directed advertising is enabled. The host does not issue this command when advertising is enabled in the controller; if it is, the command disallowed error code is used.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.5].

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 ms) ... 0x4000 (10240 ms)
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 ms	0x0020 (20 ms) ... 0x4000 (10240 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	• 0x00: Public device address • 0x01: Random device address

Parameter	Size	Description	Possible values
		• 0x01: Random device address • 0x02: Controller generates resolvable private address based on the local IRK from resolving list. If there is no matching entry, use public address. • 0x03: Controller generates resolvable private address based on the local IRK from resolving list. If there is no matching entry, use random address from LE_Set_Random_Address.	• 0x02: Resolvable private or public address • 0x03: Resolvable private or random address
Peer_Address_Type	1	Address type of the peer device	• 0x00: Public device address • 0x01: Random device address
Peer_Address	6	Public device, random device, public identity or random (static) identity address of the device to connect	-
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

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_PHYSICAL_CHANNEL_TX_POWER

This command is used by the host to read the transmit power level used for LE advertising channel packets. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.6].

Input parameters: none

Table 30. HCI_LE_READ_ADVERTISING_PHYSICAL_CHANNEL_TX_POWER output 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

This command sets the data used in advertising packets with a data field. Only the significant part of the Advertising_Data is transmitted in the packets. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.7].

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 formatted data	-

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

This command is used to provide data used in scanning packets with a data field. Only the significant part of the Scan_Response_Data is transmitted in the packets. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.8]

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 formatted data	-

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_ADVERTISING_ENABLE

This command is used to request the controller to start or stop advertising. The controller manages the timing of advertisements according to the parameters given in the HCI_LE_SET_ADVERTISING_PARAMETERS command, and continues advertising until the host issues an HCI_LE_SET_ADVERTISING_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 disabled. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.9].

Table 35. HCI_LE_SET_ADVERTISING_ENABLE input parameters

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

Table 36. **HCI_LE_SET_ADVERTISING_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

This command is used to set the scan parameters.

The LE_Scan_Type parameter controls the type of scan. 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 always set to a value smaller than or equal to the value set for the LE_Scan_Interval parameter. If they are set to the same value, scanning runs continuously. The Own_Address_Type parameter determines the address (public or random device) used 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.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.10].

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	Time interval from when the controller started its last LE scan until it begins the next one. Time = N * 0.625 ms	• 0x0004 (2.5 ms) ... 0x4000 (10240.0 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 (2.5 ms) ... 0x4000 (10240.0 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: accepts all advertisement packets. Directed advertising packets not addressed to this device are ignored. • 0x01: ignores advertisement packets from devices not in the white list. Directed advertising packets not addressed to this device are ignored • 0x02: accepts all undirected advertisement packets. Directed advertisement packets where initiator address is a RPA and directed advertisement packets addressed to this device are accepted. • 0x03: accepts all undirected advertisement packets from devices in the white list. Directed advertisement packets where initiator address is RPA and directed advertisement packets addressed to this device are 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)

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

This command is used to start scanning, 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 received packet. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.11].

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

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

This 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 not used, the Peer_Address_Type and the Peer_Address parameters specify the type and the 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 ms) is larger than (1 + Conn_Latency) * Conn_Interval_Max * 2 (Conn_Interval_Max is given in ms). 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.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.12].

Table 41. HCI_LE_CREATE_CONNECTION input parameters

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	Time from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	0x0004 (2.5 ms) ... 0x4000 (10240.0 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 (2.5 ms) ... 0x4000 (10240.0 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	Peripheral 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 + connPeripheralLatency) * 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)

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_LE_CONNECTION_COMPLETE_EVENT
- HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT

2.1.26 HCI_LE_CREATE_CONNECTION_CANCEL

This command is used to cancel the HCI_LE_CREATE_CONNECTION command, hence it issued only after it, a command status event has been received, and before the HCI_LE_CONNECTION_COMPLETE_EVENT. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.13].

Input parameters: none

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
- HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT

2.1.27 HCI_LE_READ_FILTER_ACCEPT_LIST_SIZE

This command is used to read the total number of white list entries that can be stored in the controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.14].

Input parameters: none

Table 44. **HCI_LE_READ_FILTER_ACCEPT_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_FILTER_ACCEPT_LIST

This command is used to clear the white list stored in the controller. It 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

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.15].

Input parameters: none

Table 45. **HCI_LE_CLEAR_FILTER_ACCEPT_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_FILTER_ACCEPT_LIST

This command is used to add a single device to the white list stored in the controller. It 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

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.16].

Table 46. **HCI_LE_ADD_DEVICE_TO_FILTER_ACCEPT_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.	-

Table 47. **HCI_LE_ADD_DEVICE_TO_FILTER_ACCEPT_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_FILTER_ACCEPT_LIST

This command is used to remove a single device from the white list stored in the controller. It 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

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.17].

Table 48. **HCI_LE_REMOVE_DEVICE_FROM_FILTER_ACCEPT_LIST input parameters**

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

Table 49. **HCI_LE_REMOVE_DEVICE_FROM_FILTER_ACCEPT_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

This command, supported only on central side, is used to change the link layer parameters of a connection:

- 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 ms, is larger than (1 + Conn_Latency) * Conn_Interval_Max (in ms) * 2.
- The Minimum_CE_Length and Maximum_CE_Length provide the controller information 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.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.18].

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, less than or equal to Conn_Interval_Max. Time = N * 1.25 ms	0x0006 (7.5 ms) ... 0x0C80 (4000.0 ms)
Conn_Interval_Max	2	Maximum value for the connection event interval, greater than or equal to Conn_Interval_Min. Time = N * 1.25 ms	0x0006 (7.5 ms) ... 0x0C80 (4000.0 ms)
Conn_Latency	2	Peripheral 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 + connPeripheralLatency) * 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)

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

This command, used only when the local device supports the central role, allows the host to specify a channel classification for data channels based on its local information. The classification persists until overwritten with a subsequent command, or until the controller is reset using the Reset command. If this command is used, the host must send it within 10 s after it has known that the channel classification has changed. The interval between two sent commands must be at least 1 s. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.19].

Table 52. HCI_LE_SET_HOST_CHANNEL_CLASSIFICATION input parameters

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

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

This 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 central has received or not an acknowledgement. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.20].

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

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 unused = 0. Channel n used = 1. The most significant bits are reserved, and set to 0.	-

Events generated

HCI_COMMAND_COMPLETE_EVENT

2.1.34 HCI_LE_READ_REMOTE_FEATURES

This command requests a list of the used LE features from the remote device, and returns a list of the used LE features. It can be issued on both the client and the server. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.21].

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

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

This command is used to request the controller to encrypt the Plaintext_Data using the key provided by the command, and returns the Encrypted_Data to the host. The AES-128 bit block cypher is defined in NIST Publication FIPS-197 (http://csrc.nist.gov). See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.22].

Table 58. **HCI_LE_ENCRYPT input parameters**

Parameter	Size	Description	Possible values
Key	16	128-bit key for the encryption of data given in the command	-
Plaintext_Data	16	128-bit data block to encrypt	-

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

This command is used to request the controller to generate eight octets of random data to send to the host. Random_Number is generated according to [Vol 2] Part H, Section 2 if the LE feature (LL encryption) is supported. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.23].

Input parameters: none

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_ENABLE_ENCRYPTION

This command is used to authenticate the encryption key associated with the remote device specified by the connection handle, and, once authenticated, encrypts the connection. If the connection is already encrypted the controller pauses before attempting to authenticate the given encryption key, and then re-encrypts the connection. While encryption is paused, no user data are transmitted. On an authentication failure, the connection is automatically disconnected by the link layer. If this command succeeds, the connection is encrypted. This command is used only when the local device role is central. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.24].

Table 61. HCI_LE_ENABLE_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	-

Table 62. HCI_LE_ENABLE_ENCRYPTION output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_ENCRYPTION_CHANGE_EVENT
- HCI_ENCRYPTION_KEY_REFRESH_COMPLETE_EVENT

2.1.38 HCI_LE_LONG_TERM_KEY_REQUEST_REPLY

This 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. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.25].

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	-

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_REQUEST_NEGATIVE_REPLY

This 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. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.26].

Table 65. **HCI_LE_LONG_TERM_KEY_REQUEST_NEGATIVE_REPLY input parameters**

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

Table 66. **HCI_LE_LONG_TERM_KEY_REQUEST_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

This command reads the states and state combinations supported by the link layer. LE_States is an 8-octet bit field. If a bit is set to 1, this state or state combination is supported by the controller. Multiple bits in LE_States can be set to 1, to indicate support for multiple state and state combinations.

The advertising type and the initial state combinations are set only if the corresponding advertising types and central role combination are set. The scanning types and the initial state combinations are set only if the corresponding scanning types and central role combination are set.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.27].

Input parameters: none

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

This command allows the host to suggest maximum transmission packet size and maximum packet transmission time (connMaxTxOctets and connMaxTxTime) for a given connection. The controller can use different values, based on local information. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.33].

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

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

With this command the host reads the suggested values (SuggestedMaxTxOctets and SuggestedMaxTxTime) for the controller maximum transmitted number of payload octets and maximum packet transmission time, for new connections. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.34].

Input parameters: none

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 suggested value for the controller maximum transmitted number of payload octets, to use for new connections	0x001B ... 0x00FB
SuggestedMaxTxTime	2	The host suggested value for the controller maximum packet transmission time, to use for new connections	0x0148 ... 0x4290

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.43 HCI_LE_WRITE_SUGGESTED_DEFAULT_DATA_LENGTH

With this command the host indicates to the controller values for the maximum transmission number of payload octets and maximum packet transmission time, for new connections. The controller can use different values for connInitialMaxTxOctets and connInitialMaxTxTime, based on local information. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.35].

Table 71. HCI_LE_WRITE_SUGGESTED_DEFAULT_DATA_LENGTH input parameters

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

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

This command is used to return the local P-256 public key from the controller, which generates a new P- 256 public/private key pair upon receipt of the command. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.36].

Input parameters: none

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

This command is used to generate a Diffie-Hellman key in the controller for use over the LE transport, taking the remote P-256 public key as input. The generation uses the private key from the HCI_LE_READ_LOCAL_P256_PUBLIC_KEY command. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.37].

Table 74. HCI_LE_GENERATE_DHKEY input parameters

Parameter	Size	Description	Possible values
Remote_P256_Public_Key	64	Remote P-256 public key: X, Y format octets • 31-0: X coordinate octets • 63-32: Y coordinate little endian format	-

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

This command adds a device to the list of address translations used to resolve resolvable private addresses in the controller. It cannot be used when address translation is enabled in the controller and:

- advertising is enabled
- scanning is enabled
- HCI_LE_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).

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.38].

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	-

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

This command is used to remove a device from the list of address translations used to resolve private addresses in the controller. It cannot be used when address translation is enabled in the controller and:

- advertising is enabled
- scanning is enabled
- HCI_LE_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 responds with error code 0x02 (unknown connection identifier).

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.39].

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	-

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

This command is used to remove all devices from the list of address translations used to resolve resolvable private addresses in the controller. It cannot be used when address translation is enabled in the controller and:

- advertising is enabled
- scanning is enabled
- HCI_LE_CREATE_CONNECTION command is outstanding

This command can be used at any time when address translation is disabled in the controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.40].

Input parameters: none

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

This command is used to read the total number of address translation entries in the resolving list that can be stored in the controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.41].

Input parameters: none

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

This command is used to get the current peer resolvable private address used for the corresponding peer public and random (static) identity address. The used peer resolvable address can change after the command is called. This command can be 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). See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.42].

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	-

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

This command is used to get the current local resolvable private address used for the corresponding peer identity address. The used local resolvable address can change after the command is called. This command can be 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). See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.43].

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	-

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

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

- advertising is enabled
- acanning is enabled
- HCI_LE_CREATE_CONNECTION command is outstanding

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.44].

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 disabled (default) • 0x01: address resolution enabled

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

This command sets the length of time the controller uses a resolvable private address, before a new resolvable private address is generated and starts to be used. This timeout applies to all addresses generated by the controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.45].

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 0x0001 to 0xA1B8 (1 s to ~11.5 hours), default 0x0384 (900 s)	-

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

This command allows the host to read the controller maximum supported payload octets and packet duration times for transmission and reception (supportedMaxTxOctets, supportedMaxTxTime, supportedMaxRxOctets, and supportedMaxRxTime). See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.46].

Input parameters: none

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

This command is used to read the current transmitter and receiver PHY on the connection identified by the Connection_Handle. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.47].

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

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

This command allows the host to specify its preferred values for the transmitter and receiver PHY, to use for all subsequent connections over the LE transport.

ALL_PHYS parameter is allows the host to specify, for each direction, whether it has no preferences among the PHYs that the controller supports in a given direction, or whether there is a particular PHY that it prefers in the TX_PHYS or RX_PHYS parameter. TX_PHYS parameter indicates the transmitter PHYs that the host prefers the controller to use. If ALL_PHYS specifies that the host has no preference, TX_PHYS parameter is ignored, otherwise at least one bit is set to 1. RX_PHYS parameter indicates the receiver PHYs that the host prefers the controller to use. If ALL_PHYS specifies that the host has no preference, RX_PHYS is ignored, otherwise at least one bit is set to 1.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.48].

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	• 0: the host has no preferences among the transmitter PHYs supported by the controller • 1: the host has no preferences among the receiver PHYs supported by the controller
TX_PHYS	1	Host preferences for TX PHY	Bitmask of: • 0x01: LE 1M PHY- preferred • 0x02: LE 2M PHY- preferred • 0x04: LE coded PHY-preferred (not supported by STM32WB)
RX_PHYS	1	Host preferences for RX PHY	Bitmask of: • 0x01: LE 1M PHY- preferred • 0x02: LE 2M PHY- preferred • 0x04: LE coded PHY-preferred (not supported by STM32WB)

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

This command is used to set the PHY preferences for the connection identified by the Connection_Handle. The controller can be unable to make the change (for example, when the peer does not support the requested PHY), or decide that the current PHY is the preferred one.

The ALL_PHYS parameter 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 there is a PHY that it prefers in the TX_PHYS or RX_PHYS parameter. The TX_PHYS/RX_PHYS parameters indicate to the transmitter/receiver PHYs that the host prefers the controller to use. If the ALL_PHYS parameter specifies that the Host has no preference, the TX_PHYS/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 among the preferred, the controller can request a change. The PHY preferences provided by the command override those provided via the HCI_LE_SET_DEFAULT_PHY command or any preferences previously set using this 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 0. The controller can override any preferred coding for transmitting on the LE coded PHY. The host can specify a preferred coding, even if it prefers not to use the LE coded transmitter PHY, as the controller can override the PHY preference.

See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.49].

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	Bitmask of: • 0x01: LE 1M PHY preferred • 0x02: LE 2M PHY preferred • 0x04: LE Coded PHY preferred (not supported by STM32WB)
TX_PHYS	1	Host preferences for TX PHY	Bitmask of: • 0x01: LE 1M PHY preferred • 0x02: LE 2M PHY preferred • 0x04: LE Coded PHY preferred (not supported by STM32WB)
RX_PHYS	1	Host preferences for RX PHY	Bitmask of: • 0x01: LE 1M PHY preferred • 0x02: LE 2M PHY preferred • 0x04: LE Coded PHY preferred (not supported by STM32WB)
PHY_options	2	Options for PHYs (not supported by STM32WB)	• 0x0000: the host has no preferred coding when transmitting on the LE coded PHY • 0x0001: the host prefers S = 2 coding when transmitting on the LE coded PHY • 0x0002: the host prefers S = 8 coding when transmitting on the LE coded PHY

Table 96. HCI_LE_SET_PHY output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT

2.1.58 HCI_LE_SET_ADVERTISING_SET_RANDOM_ADDRESS

This command is used by the Host to set the random device address specified by the Random_Address parameter. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.52].

Table 97. HCI_LE_SET_ADVERTISING_SET_RANDOM_ADDRESS input parameters

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 … 0xEF
Random_Address	6	Random device address	-

Table 98. HCI_LE_SET_ADVERTISING_SET_RANDOM_ADDRESS output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.59 HCI_LE_SET_EXTENDED_ADVERTISING_PARAMETERS

This command is used by the Host to set the extended advertising parameters. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.53].

Table 99. HCI_LE_SET_EXTENDED_ADVERTISING_PARAMETERS input parameters

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 … 0xEF
Adv_Event_Properties	2	Type of advertising event.	Bitmask of: • 0x0001: Connectable advertising • 0x0002: Scannable advertising • 0x0004: Directed advertising • 0x0008: High duty-cycle directed connectable advertising • 0x0010: Use legacy advertising PDUs • 0x0020:Anonymous advertising • 0x0040: Include Tx Power in at least one advertising PDU
Primary_Adv_Interval_Min	3	Minimum advertising interval. Time = N * 0.625 ms	0x000020 (20.000 ms) ... 0xFFFFFF (10485759.375 ms)
Primary_Adv_Interval_Max	3	Maximum advertising interval. Time = N * 0.625 ms	0x000020 (20.000 ms) ... 0xFFFFFF (10485759.375 ms)
Primary_Adv_Channel_Map	1	Advertising channel map	Bitmask of: • 0x01: Use Channel 37 • 0x02: Use Channel 38 • 0x04: Use Channel 39
Own_Address_Type	1	Own address type	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address if available, otherwise Public address • 0x03: Resolvable private address if available, otherwise Random address
Peer_Address_Type	1	Address type of the peer device	• 0x00: Public device address or Public identity address • 0x01: Random device address or 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.	-
Adv_Filter_Policy	1	Advertising filter policy	• 0x00: Process scan and connection requests from all devices (White list not in use) • 0x01: Process connection requests from all devices and scan requests only from devices in the White list. • Process scan requests from all devices andconnection requests only from devices in the White list. • 0x03: Process scan and connection requests only from devices in the White List.
Adv_TX_Power	1	Advertising TX power. Units: dBm.	• -127 ... 20 • 127: Host has no preferences

Parameter	Size	Description	Possible values
Primary_Adv_PHY	1	Primary advertising PHY	• 0x01: Primary advertisement PHY is LE 1M • 0x03:Primary advertisement PHY is LE coded (not supported by STM32WB devices)
Secondary_Adv_Max_Skip	1	Secondary advertising maximum skip	• 0x00: AUX_ADV_IND must be sent prior to the next advertising event • 0x01 ... 0xFF: Maximum advertising events that the controller can skip before sending the AUX_ADV_IND packets on the secondary advertising physical channel
Secondary_Adv_PHY	1	Secondary advertising PHY	• 0x01: Secondary advertisement PHY is LE 1M • 0x02: Secondary advertisement PHY is LE 2M • 0x03: Secondary advertisement PHY is LE coded (not supported by STM32WB devices)
Adv_SID	1	Value of the Advertising SID subfield in the ADIfield of the PDU	• 0x00 ... 0x0F
Scan_Req_Notification_Enable	1	Scan request notifications	• 0x00: Scan request notifications disabled • 0x01:Scan request notifications enabled

Table 100. HCI_LE_SET_EXTENDED_ADVERTISING_PARAMETERS output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Selected_TX_Power	1	Power level selected by the controller. Units: dBm.	• -127 ... 20

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.60 HCI_LE_SET_EXTENDED_ADVERTISING_DATA

This command is used to set the data used in extended advertising PDUs with a data field. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.54].

Table 101. HCI_LE_SET_EXTENDED_ADVERTISING_DATA input parameters

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 … 0xEF
Operation	1	Advertising operation	• 0x00: intermediate fragment of fragmented extended advertising data • 0x01: first fragment of fragmented extended advertising data • 0x02: last fragment of fragmented extended advertising data • 0x03: complete extended advertising data • 0x04: unchanged data (just update the advertising DID)
Fragment_Preference	1	Fragment preference	• 0x00: the controller can fragment all data • 0x02: the controller should not fragment data, or minimize fragmentation

Parameter	Size	Description	Possible values
Advertising_Data_Length	1	Length of Advertising_Data in octets	-
Advertising_Data	Advertising_Data_Length	Data formatted as defined in Bluetooth spec. v.5.2 [Vol 3, Part C, 11].	-

Table 102. HCI_LE_SET_EXTENDED_ADVERTISING_DATA output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.61 HCI_LE_SET_EXTENDED_SCAN_RESPONSE_DATA

This command is used by to provide scan response data used in scanning response PDUs during extended advertising. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.55].

Table 103. HCI_LE_SET_EXTENDED_SCAN_RESPONSE_DATA input parameters

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 … 0xEF
Operation	1	Scan response operation	• 0x00: intermediate fragment of fragmented scan response data • 0x01: first fragment of fragmented scan response data • 0x02: last fragment of fragmented scan response data • 0x03: complete scan response data
Fragment_Preference	1	Fragment preference	• 0x00: the controller can fragment all data • 0x02: the controller should not fragment data, or minimize fragmentation
Scan_Response_Data_Length	1	Length of Scan_Response_Data in octets	-
Scan_Response_Data	Scan_Response_Data_Length	Data formatted as defined in Bluetooth spec. v.5.2 [Vol 3, Part C, 11].	-

Table 104. HCI_LE_SET_EXTENDED_SCAN_RESPONSE_DATA output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.62 HCI_LE_SET_EXTENDED_ADVERTISING_ENABLE

This command is used to request the controller to enable/disable one or more advertising sets using those identified by the Advertising_Handle[i] parameter. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.56].

Table 105. HCI_LE_SET_EXTENDED_ADVERTISING_ENABLE input parameters

Parameter	Size	Description	Possible values
Enable	1	Enable/disable advertising	• 0x00: advertising disabled • 0x01: advertising enabled
Num_Sets	1	Number of advertising sets	• 0x00: disable all advertising sets • 0x00 … 0x3F: number of advertising sets to enable/disable
Advertising_Handle[i]	1	Used to identify an advertising set	0x00 … 0xEF
Duration[i]	2	Duration of advertising set. Time = N * 10 ms.	• 0x0000 (0 ms): no advertising duration • 0x0001 (10 ms) ... 0xFFFF (655350 ms): advertising duration
Max_Extended_Advertising_Events[i]	1	Maximum number of advertising events	• 0x00 (0 ms): no maximum number • 0x01 ... 0xFF: maximum number of events the controller must try to send before terminating the extended advertising

Table 106. HCI_LE_SET_EXTENDED_ADVERTISING_ENABLE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT

2.1.63 HCI_LE_READ_MAXIMUM_ADVERTISING_DATA_LENGTH

This command is used by the Host to set the random device address specified by the Random_Address parameter. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.57].

Input parameters: none

Table 107. HCI_LE_READ_MAXIMUM_ADVERTISING_DATA_LENGTH output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Max_Advertising_Data_Length	2	Maximum supported advertising data length	0x001F ... 0x0672

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.64 HCI_LE_READ_NUMBER_OF_SUPPORTED_ADVERTISING_SETS

This command is used by the Host to set the random device address specified by the Random_Address parameter. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.58].

Input parameters: none

Table 108. **HCI_LE_READ_NUMBER_OF_SUPPORTED_ADVERTISING_SETS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Num_Supported_Advertising_Sets	1	Number of advertising sets supported at the same time	0x01 ... 0xF0

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.65 HCI_LE_REMOVE_ADVERTISING_SET

This command is used to remove an advertising set from the controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.59].

Table 109. **HCI_LE_REMOVE_ADVERTISING_SET input parameters**

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 … 0xEF

Table 110. **HCI_LE_REMOVE_ADVERTISING_SET output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.66 HCI_LE_CLEAR_ADVERTISING_SETS

This command is used to remove all existing advertising sets from the controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.60].

Input parameters: none

Table 111. **HCI_LE_CLEAR_ADVERTISING_SETS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.67 HCI_LE_SET_EXTENDED_SCAN_PARAMETERS

This command is used to set the extended scan parameters to be used on the advertising physical channels. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.64].

Table 112. **HCI_LE_SET_EXTENDED_SCAN_PARAMETERS input parameters**

Parameter	Size	Description	Possible values
Own_Address_Type	1	Own address type	• 0x00: public device address • 0x01: random device address • 0x02: resolvable private address if available, otherwise public address • 0x03: resolvable private address if available, otherwise random address

Parameter	Size	Description	Possible values
Scanning_Filter_Policy	1	Scan filter policy	• 0x00: accept all advertising and scan response PDUs, except directed advertising PDUs not addressed to this device. • 0x01: accept only advertising and scan response PDUs from devices where the advertiser address is in the white list. Directed advertising PDUs not addressed to this device are ignored. • 0x02: accept all advertising and scan response PDUs, except directed advertising PDUs where the identity address corresponding to TargetA does not address this device. • 0x03: accept all advertising and scan response PDUs, except advertising and scan response PDUs where the advertiser identity address is not in the white list, and directed advertising PDUs where the identity address corresponding to TargetA does not address this device.
Scanning_PHYs	1	Scan PHYs	Bitmask of: • 0x01: scan advertisements on the LE 1M PHY • 0x04: scan advertisements on the LE Coded PHY (not supported by STM32WB devices)
Scan_Type	1	Passive or active scanning. With passive scanning no scan request PDUs are sent.	• 0x00: passive scanning • 0x01: active scanning
Scan_Interval	2	Time interval from when the controller started its last scan until it begins the subsequent scan on the primary advertising physical channel. Time = N * 0.625 ms.	0x0004 (2.5 ms) ... 0x5DC0 (15000.0 ms)
Scan_Window	2	Duration of the scan on the primary advertising physical channel. Time = N * 0.625 ms.	0x0004 (2.5 ms) ... 0x5DC0 (15000.0 ms)

Table 113. HCI_LE_SET_EXTENDED_SCAN_PARAMETERS output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.68 HCI_LE_SET_EXTENDED_SCAN_ENABLE

This command is used to enable/disable extended scanning. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.65].

Table 114. HCI_LE_SET_EXTENDED_SCAN_ENABLE input parameters

Parameter	Size	Description	Possible values
Enable	1	Enable/disable scan	• 0x00: scanning disabled • 0x01: scanning enabled
Filter_Duplicates	1	Duplicate filtering	• 0x00: duplicate filtering disabled • 0x01: duplicate filtering enabled • 0x02: duplicate filtering enabled, reset for each scan period

Parameter	Size	Description	Possible values
Duration	2	Scan duration. Time = N * 10 ms.	• 0x0000 (0 ms): scan continuously until explicitly disabled • 0x0001 (10 ms) ... 0xFFFF (655350 ms)
Period	2	Scan period. Time = N * 1.28 s.	• 0x0000 (0 ms): scan continuously • 0x0001 (1280 ms) ... 0xFFFF (83884800 ms): time interval from when the controller started its last Scan_Duration until it begins the subsequent one

Table 115. HCI_LE_SET_EXTENDED_SCAN_ENABLE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT

2.1.69 HCI_LE_EXTENDED_CREATE_CONNECTION

This command is used to create an ACL connection to a connectable advertiser by means of extended scanning. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.66].

Table 116. HCI_LE_EXTENDED_CREATE_CONNECTION input parameters

Parameter	Size	Description	Possible values
Initiator_Filter_Policy	1	Initiator filter policy	• 0x00: white list not used • 0x01: white list used
Own_Address_Type	1	Own address type	• 0x00: public device • 0x01: random device • 0x02: resolvable private address if available, otherwise public address • 0x03: resolvable public address if available, otherwise random address
Peer_Address_Type	1	Address type of the peer device	• 0x00: public device or public identity address • 0x01: random device or random (static) identity address
Peer_Address	6	Public or random device address, public or random (static) identity address of the device to connect	-
Initiating_PHYs	1	Initiating PHYs	Bitmask of: • 0x01: scan connectable advertisements on the LE 1M PHY connection parameters for the LE 1M PHY • 0x02: connection parameters for the LE 2M PHY • 0x04: scan connectable advertisements on the LE coded PHY (not supported on STM32WB)
Scan_Interval	2	Time interval from when the controller started its last scan until it begins the subsequent scan on the primary advertising physical channel. Time = N * 0.625 ms.	0x0004 (2.5 ms) ... 0x5DC0 (15000.0 ms)
Scan_Window	2	Duration of the scan on the primary advertising physical channel. Time = N * 0.625 ms.	0x0004 (2.5 ms) ... 0x5DC0 (15000.0 ms)

Parameter	Size	Description	Possible values
Conn_Interval_Min	2	Minimum value of the connection event interval (must be lower than or equal to Conn_Interval_Max. Time = N * 1.25 ms).	0x0006 (7.5 ms) ... 0x0C80 (4000.0 ms)
Conn_Interval_Max	2	Maximum value of the connection event interval (must be higher than or equal to Conn_Interval_Min. Time = N * 1.25 ms).	0x0006 (7.5 ms) ... 0x0C80 (4000.0 ms)
Conn_Latency	2	Peripheral latency for the connection, in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE Link. Must be a multiple of 10 ms and larger than (1 + connPeripheralLatency) * connInterval * 2. Time = N * 10 ms.	0x000A (100 ms) ... 0x0C80 (32000 ms)
Min_CE_Length	2	Minimum length needed for this LE connection. Time = N * 0.625 ms.	0x0000 (0.0 ms) ... 0xFFFF (40959.375 ms)
Max_CE_Length	2	Maximum length needed for this LE connection. Time = N * 0.625 ms.	0x0000 (0.0 ms) ... 0xFFFF (40959.375 ms)

Table 117. HCI_LE_EXTENDED_CREATE_CONNECTION output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- HCI_LE_CONNECTION_COMPLETE_EVENT

2.1.70 HCI_LE_READ_TRANSMIT_POWER

This command is used to read the minimum and maximum transmit power supported by the controller. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.74].

Input parameters : none

Table 118. HCI_LE_READ_TRANSMIT_POWER output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Min_TX_Power	1	Signed integer, in dBm	-127 ... 20
Max_TX_Power	1	Signed integer, in dBm	-127 ... 20

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.71 HCI_LE_READ_RF_PATH_COMPENSATION

This command is used to read the RF path compensation value parameters used in the Tx power level and RSSI calculation. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.75].

Input parameters: none

Table 119. **HCI_LE_READ_RF_PATH_COMPENSATION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
RF_TX_Path_Compensation	2	RF TX path compensation value (16-bit signed integer), in 0.1 dB units	-1280 … 1280
RF_RX_Path_Compensation	2	RF RX path compensation value (16-bit signed integer), in 0.1 dB units	-1280 … 1280

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.72 HCI_LE_WRITE_RF_PATH_COMPENSATION

This command is used to indicate the contribution of intermediate components in the RF path to the gain or loss between the RF transceiver and the antenna. Positive/negative values mean a net RF path gain/loss. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.76].

Table 120. **HCI_LE_WRITE_RF_PATH_COMPENSATION input parameters**

Parameter	Size	Description	Possible values
RF_TX_Path_Compensation	2	RF TX path compensation value (16-bit signed integer), in 0.1 dB units	-1280 … 1280
RF_RX_Path_Compensation	2	RF RX path compensation value (16-bit signed integer), in 0.1 dB units	-1280 … 1280

Table 121. **HCI_LE_WRITE_RF_PATH_COMPENSATION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.73 HCI_LE_SET_PRIVACY_MODE

This command is used to allow the host to specify the privacy mode to use for a given entry on the resolving list. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.74].

Table 122. **HCI_LE_SET_PRIVACY_MODE 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	-
Privacy Mode	1	Privacy mode	• 0x00: Use network privacy mode • 0x01: Use device privacy mode

Table 123. **HCI_LE_SET_PRIVACY_MODE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.1.74 HCI_LE_GENERATE_DHKEY_V2

This command is used to start the generation of a Diffie-Hellman key in the controller for use over the LE transport. It takes the remote P-256 public key as input, and uses the private key generated by the HCI_LE_READ_LOCAL_P256_PUBLIC_KEY command or the private debug key. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.37].

Table 124. HCI_LE_GENERATE_DHKEY_V2 input parameters

Parameter	Size	Description	Possible values
Remote_P256_Public_Key	64	The remote P-256 public key in X, Y format: • Octets 31-0: X coordinate • Octects 63-32: Y coordinate, little Endian format	-
Key_Type	1	Type of private key used for the Diffie-Hellman key generation	• 0x00: Use the generated private key • 0x01: Use the debug private key

Table 125. HCI_LE_GENERATE_DHKEY_V2 output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- HCI_LE_GENERATE_DHKEY_COMPLETE_EVENT

2.2 HCI testing commands

In Table 126 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / PO / BF/ LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 126. HCI testing commands list

Command	OpCode	LO	PO	BO	BF	LB
HCI_LE_RECEIVER_TEST	0x201D	Y	Y	Y	Y	Y
HCI_LE_TRANSMITTER_TEST	0x201E	Y	Y	Y	Y	Y
HCI_LE_TEST_END	0x201F	Y	Y	Y	Y	Y
HCI_LE_RECEIVER_TEST_V2	0x2033	Y	-	-	Y	Y
HCI_LE_TRANSMITTER_TEST_V2	0x2034	Y	-	-	Y	Y

2.2.1 HCI_LE_RECEIVER_TEST

This command is used to start a test where the DUT receives test reference packets at a fixed interval from the tester. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.28].

Table 127. HCI_LE_RECEIVER_TEST input parameters

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

Table 128. 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

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 command supports Packet_Payload values 0x00, 0x01, and 0x02. An LE controller supports other values. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.29].

Table 129. HCI_LE_TRANSMITTER_TEST input parameters

Parameter	Size	Description	Possible values
TX_Frequency	1	N = (F - 2402) / 2 - Frequency range: 2402 to 2480 MHz	0x00 ... 0x27
Length_Of_Test_Data	1	Length in bytes of payload data in each packet	• 0x00 ... 0x25: BO variant • 0x00 … 0xFF: otherwise
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

Table 130. **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

This command is used to stop any test in progress. The Number_Of_Packets is an unsigned number and contains the number of received packets, for a transmitter test is 0x0000. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.30].

Input parameters: none

Table 131. **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_RECEIVER_TEST_V2

This command is used to start a test where the DUT receives test reference packets at a fixed interval. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.28].

Table 132. **HCI_LE_RECEIVER_TEST_V2 input parameters**

Parameter	Size	Description	Possible values
RX_Frequency	1	N = (F - 2402) / 2 - Frequency range: 2402 to 2480 MHz	0x00 ... 0x27
PHY	1	PHY to be used by the receiver	• 0x01: Transmitter set to use the LE 1M PHY • 0x02: Transmitter set to use the LE 2M PHY • 0x03: Receiver set to use the LE coded PHY (not supported by STM32WB devices)
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

Table 133. **HCI_LE_RECEIVER_TEST_V2 output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.2.5 HCI_LE_TRANSMITTER_TEST_V2

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 this 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 MCUs). An LE controller may support other values of Packet_Payload. See Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.29].

Table 134. HCI_LE_TRANSMITTER_TEST_V2 input parameters

Parameter	Size	Description	Possible values
TX_Frequency	1	N = (F - 2402) / 2 - Frequency range: 2402 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

Table 135. HCI_LE_TRANSMITTER_TEST_V2 output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3 HAL commands

In Table 136 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 136. HAL commands list

Command	OpCode	LO	PO	BO	BF	LB
ACI_HAL_GET_FW_BUILD_NUMBER	0xFC00	Y	Y	Y	Y	Y
ACI_HAL_WRITE_CONFIG_DATA	0xFC0C	Y	Y	Y	Y	Y
ACI_HAL_READ_CONFIG_DATA	0xFC0D	Y	Y	Y	Y	Y
ACI_HAL_SET_TX_POWER_LEVEL	0xFC0F	Y	Y	Y	Y	Y
ACI_HAL_LE_TX_TEST_PACKET_NUMBER	0xFC14	Y	Y	-	Y	Y
ACI_HAL_TONE_START	0xFC15	Y	Y	Y	Y	Y
ACI_HAL_TONE_STOP	0xFC16	Y	Y	Y	Y	Y
ACI_HAL_GET_LINK_STATUS	0xFC17	Y	Y	-	Y	Y
ACI_HAL_SET_RADIO_ACTIVITY_MASK	0xFC18	Y	Y	Y	Y	Y
ACI_HAL_GET_ANCHOR_PERIOD	0xFC19	Y	Y	Y	Y	Y
ACI_HAL_SET_EVENT_MASK	0xFC1A	-	-	-	-	-
ACI_HAL_GET_PM_DEBUG_INFO	0xFC1C	-	-	-	-	-
ACI_HAL_SET_PERIPHERAL_LATENCY	0xFC20	Y	-	-	Y	Y
ACI_HAL_READ_RSSI	0xFC22	Y	Y	Y	Y	Y
ACI_HAL_EAD_ENCRYPT_DECRYPT	0xFC2F	-	-	-	-	-
ACI_HAL_READ_RADIO_REG	0xFC30	-	-	-	-	-
ACI_HAL_WRITE_RADIO_REG	0xFC31	-	-	-	-	-
ACI_HAL_READ_RAW_RSSI	0xFC32	-	-	-	-	-
ACI_HAL_RX_START	0xFC33	-	-	-	-	-
ACI_HAL_RX_STOP	0xFC34	-	-	-	-	-
ACI_HAL_STACK_RESET	0xFC3B	Y	Y	Y	Y	Y

2.3.1 ACI_HAL_GET_FW_BUILD_NUMBER

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

Input parameters: none

Table 137. 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

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

Note: The HCI_RESET command resets the configure data structure.

Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters

Parameter	Size	Description	Possible values
Offset	1	Offset of the element to write in the configuration data structure	• 0x00: CONFIG_DATA_PUBADDR_OFFSET: Bluetooth public address, 6 bytes • 0x08: CONFIG_DATA_ER_OFFSET: encryption root key used to derive LTK and CSRK, 16 bytes • 0x18: CONFIG_DATA_IR_OFFSET: identity root key used to derive LTK and CSRK, 16 bytes • 0x2E: CONFIG_DATA_RANDOM_ADDRESS_OFFSET (host only): static random address, 6 bytes • 0x34: CONFIG_DATA_GAP_ADD_REC_NBR_OFFSET: GAP service additional record number • 0x35: CONFIG_DATA_SC_KEY_TYPE_OFFSET (host only): secure connections key type (0: normal, 1: debug), 1 byte • 0xB0: CONFIG_DATA_SMP_MODE_OFFSET (host only): SMP mode (8-bit bitmap: bit 0: bypass, bit 1: no blaclist, bit 2: no peer debug key, bit 3: pairing request event, bit 4: no just works, bit 5: no passkey entry, bit 6: no out of band, bit 7: no numeric comparison), 1 byte • 0xC0: CONFIG_DATA_LL_SCAN_CHAN_MAP_OFFSET (only for STM32WB devices): LL scan channel map (same format as Primary_Adv_Channel_Map), 1 byte • 0xC1: CONFIG_DATA_LL_BG_SCAN_MODE_OFFSET (only for STM32WB devices): LL background scan mode (0: BG scan disabled, 1: BG scan enabled), 1 byte • 0xD1: CONFIG_DATA_LL_MAX_DATA_EXT_OFFSET (only for STM32WB full stack); LL maximum data length extension (bytes 0-1: supportedMaxTxOctets, bytes 2-3: supportedMaxTxTime, bytes 4-5: supportedMaxRxOctets, bytes 6-7: supportedMaxRxTime); 8 bytes
Length	1	Length of data to write	-
Value	Length	Data to write	-

Table 139. 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

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

Table 140. ACI_HAL_READ_CONFIG_DATA input parameters

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

Table 141. **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

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, and use this command to observe the change of signal strength. The system keeps the last received power level from the command, i.e. the second command overwrites the previous power level. The new power level remains until the next ACI_HAL_SET_TX_POWER_LEVEL command, or the system reboots.

The advertising extensions commands allow, per advertising set, to override the value determined by the ACI_HAL_SET_TX_POWER_LEVEL command (see ACI_GAP_ADV_SET_CONFIGURATION). Refer to Section 5: Tx power level for the values of PA_Level parameter.

Table 142. **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.	0x00 … 0x23

Table 143. **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

This command returns the number of packets sent in direct test mode. When the direct TX test is started, a 16-bit counter is used to count the transmitted packets, starting from 0, and counting upwards. The counter can wrap and start from 0 again. The counter is not cleared until the next direct TX test starts.

Input parameters: none

Table 144. **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

This command starts a carrier frequency (a tone) on a specific channel. The frequency sine wave at the specific channel can be used only for debugging purposes. The channel ID is a parameter from 0x00 to 0x27 for the 40 BLE channels (0x00 for 2.402 GHz, 0x01 for 2.404 GHz, etc). This command cannot be used when normal Bluetooth activities are ongoing. The tone is stopped by an ACI_HAL_TONE_STOP command.

Table 145. ACI_HAL_TONE_START input parameters

Parameter	Size	Description	Possible values
RF_Channel	1	BLE channel ID, from 0x00 to 0x27, meaning (2.402 + 0.002 * 0xXX) GHz. The device continuously emits 0s, indicating 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

Table 146. 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

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

Input parameters: none

Table 147. 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

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

Input parameters: none

Table 148. 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 peripheral role • 0x03: Scanning • 0x04: Reserved • 0x05: Connected in central role • 0x06: TX test mode • 0x07: RX test mode • 0x81: Advertising with additional beacon

Parameter	Size	Description	Possible values
Link_Connection_Handle	16	Array of connection handles (two bytes) for eight links. Valid only if the link status is "connected" (0x02 or 0x05).	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.9 ACI_HAL_SET_RADIO_ACTIVITY_MASK

This command sets the bitmask associated to ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT. Only the activities enabled in the mask are reported to application by ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT.

Table 149. 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 peripheral • 0x0008: Scanning • 0x0010: Connection request • 0x0020: Connection event central • 0x0040: TX test mode • 0x0080: RX test mode

Table 150. 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

This command returns information about the Anchor Period, to help an application operating in multi-link scenarios in the selection of timing slots.

Input parameters: none

Table 151. 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

This command is used to enable/disable the generation of HAL events. If the bit in the Event_Mask is set to 1, the event associated with that bit is enabled.

Table 152. **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

Table 153. **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

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

Input parameters: none

Table 154. **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 TX	-
Allocated_For_RX	1	MBlocks allocated for RX	-
Allocated_MBlocks	1	Overall allocated MBlocks	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.13 ACI_HAL_SET_PERIPHERAL_LATENCY

This command is used to disable/enable the peripheral latency feature (by default enabled) during a connection.

Table 155. **ACI_HAL_SET_PERIPHERAL_LATENCY input parameters**

Parameter	Size	Description	Possible values
Enable	1	Enable/disable peripheral latency	• 0x00: latency disabled • 0x01: latency enabled

Table 156. **ACI_HAL_SET_PERIPHERAL_LATENCY output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.14 ACI_HAL_READ_RSSI

This command returns the RSSI value.

Input parameters: none

Table 157. **ACI_HAL_READ_RSSI output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
RSSI	1	RSSI (signed integer), in dBm	• 127: RSSI not available • -127 … 20

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.15 ACI_HAL_EAD_ENCRYPT_DECRYPT

This command encrypts or decrypts data following the Encrypted Advertising Data scheme.

When encryption mode is selected, In_Data must contain only the Payload field to encrypt. The command adds the Randomizer and MIC fields in the result. The result data length (Out_Data_Length) is equal to the input length plus 9.

When decryption mode is selected, In_Data must contain the full Encrypted Data (Randomizer + Payload + MIC). The result data length (Out_Data_Length) is equal to the input length minus 9. If the decryption fails, the returned status is BLE_STATUS_FAILED, otherwise it is BLE_STATUS_SUCCESS.

On STM32WB, the In_Data_Length value must not exceed (BLE_CMD_MAX_PARAM_LEN - 27), that is, 228 for the default value.

Table 158. **ACI_HAL_EAD_ENCRYPT_DECRYPT input parameters**

Parameter	Size	Description	Possible values
Mode	1	EAD operation mode: encryption or decryption	• 0x00: Encryption • 0x01: Decryption
Key	16	Session key used for EAD operation (in Little endian format)	-
IV	8	Initialization vector used for EAD operation (in Little endian format)	-
In_Data_Length	2	Length of input data	-
In_Data	In_Data_Length	Input data	-

Table 159. **ACI_HAL_EAD_ENCRYPT_DECRYPT output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Out_Data_Length		Length of result data	-
Out_Data	Out_Data_Length	Result data	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.16 ACI_HAL_READ_RADIO_REG

This command reads the register value from the RF module.

Table 160. **ACI_HAL_READ_RADIO_REG input parameters**

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

Table 161. **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

2.3.17 ACI_HAL_WRITE_RADIO_REG

This command writes Register value to the RF module.

Table 162. **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	-

Table 163. **ACI_HAL_WRITE_RADIO_REG output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.18 ACI_HAL_READ_RAW_RSSI

This command returns the raw value of the RSSI.

Input parameters: none

Table 164. **ACI_HAL_READ_RAW_RSSI output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Value	3	Raw RSSI value	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.19 ACI_HAL_RX_START

This command sets up the RF to listen to a specific RF channel.

Table 165. **ACI_HAL_RX_START input parameters**

Parameter	Size	Description	Possible values
RF_Channel	1	BLE channel ID, from 0x00 to 0x27 meaning (2.402 + 0.002 * 0xXX) GHz. The device continuously emits 0s, meaning that the tone is at the channel centre frequency, minus the maximum frequency deviation (250 kHz).	0x00 ... 0x27

Table 166. **ACI_HAL_RX_START output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.20 ACI_HAL_RX_STOP

This command stops a previous ACI_HAL_RX_START command.

Input parameters: none

Table 167. **ACI_HAL_RX_STOP output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.3.21 ACI_HAL_STACK_RESET

This command is equivalent to HCI_RESET, but ensures that Sleep mode is entered immediately after its completion.

Input parameters: none

Table 168. **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 169 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 169. GAP commands list

Command	OpCode	LO	PO	BO	BF	LB
ACI_GAP_SET_NON_DISCOVERABLE	0xFC81	-	Y	-	Y	-
ACI_GAP_SET_LIMITED_DISCOVERABLE	0xFC82	-	Y	-	Y	-
ACI_GAP_SET_DISCOVERABLE	0xFC83	-	Y	-	Y	-
ACI_GAP_SET_DIRECT_CONNECTABLE	0xFC84	-	Y	-	Y	-
ACI_GAP_SET_IO_CAPABILITY	0xFC85	-	Y	-	Y	-
ACI_GAP_SET_AUTHENTICATION_REQUIREMENT	0xFC86	-	Y	-	Y	-
ACI_GAP_SET_AUTHORIZATION_REQUIREMENT	0xFC87	-	Y	-	Y	-
ACI_GAP_PASS_KEY_RESP	0xFC88	-	Y	-	Y	-
ACI_GAP_AUTHORIZATION_RESP	0xFC89	-	Y	-	Y	-
ACI_GAP_INIT	0xFC8A	-	Y	-	Y	-
ACI_GAP_SET_NON_CONNECTABLE	0xFC8B	-	Y	-	Y	-
ACI_GAP_SET_UNDIRECTED_CONNECTABLE	0xFC8C	-	Y	-	Y	-
ACI_GAP_PERIPHERAL_SECURITY_REQ	0xFC8D	-	Y	-	Y	-
ACI_GAP_UPDATE_ADV_DATA	0xFC8E	-	Y	-	Y	-
ACI_GAP_DELETE_AD_TYPE	0xFC8F	-	Y	-	Y	-
ACI_GAP_GET_SECURITY_LEVEL	0xFC90	-	Y	-	Y	-
ACI_GAP_SET_EVENT_MASK	0xFC91	-	Y	-	Y	-
ACI_GAP_CONFIGURE_FILTER_ACCEPT_LIST	0xFC92	-	Y	-	Y	-
ACI_GAP_TERMINATE	0xFC93	-	Y	-	Y	-
ACI_GAP_CLEAR_SECURITY_DB	0xFC94	-	Y	-	Y	-
ACI_GAP_ALLOW_REBOND	0xFC95	-	Y	-	Y	-
ACI_GAP_START_LIMITED_DISCOVERY_PROC	0xFC96	-	-	-	Y	-
ACI_GAP_START_GENERAL_DISCOVERY_PROC	0xFC97	-	-	-	Y	-
ACI_GAP_START_AUTO_CONNECTION_ESTABLISH_PROC	0xFC99	-	-	-	Y	-
ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC	0xFC9A	-	-	-	Y	-
ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC	0xFC9B	-	-	-	Y	-
ACI_GAP_CREATE_CONNECTION	0xFC9C	-	-	-	Y	-
ACI_GAP_TERMINATE_GAP_PROC	0xFC9D	-	-	-	Y	-
ACI_GAP_START_CONNECTION_UPDATE	0xFC9E	-	-	-	Y	-
ACI_GAP_SEND_PAIRING_REQ	0xFC9F	-	-	-	Y	-
ACI_GAP_RESOLVE_PRIVATE_ADDR	0XFCA0	-	-	-	Y	-
ACI_GAP_SET_BROADCAST_MODE	0xFCA1	-	-	-	Y	-
ACI_GAP_START_OBSERVATION_PROC	0xFCA2	-	-	-	Y	-
ACI_GAP_GET_BONDED_DEVICES	0xFCA3	-	Y	-	Y	-
ACI_GAP_IS_DEVICE_BONDED	0xFCA4	-	Y	-	Y	-
ACI_GAP_NUMERIC_COMPARISON_VALUE_CONFIRM_YESNO	0xFCA5	-	Y	-	Y	-

Command	OpCode	LO	PO	BO	BF	LB
ACI_GAP_PASSKEY_INPUT	0xFCA6	-	Y	-	Y	-
ACI_GAP_GET_OOB_DATA	0xFCA7	-	Y	-	Y	-
ACI_GAP_SET_OOB_DATA	0xFCA8	-	Y	-	Y	-
ACI_GAP_ADD_DEVICES_TO_RESOLVING_LIST	0xFCA9	-	Y	-	Y	-
ACI_GAP_REMOVE_BONDED_DEVICE	0xFCAA	-	Y	-	Y	-
ACI_GAP_ADD_DEVICES_TO_LIST	0xFCAB	-	Y	-	Y	-
ACI_GAP_PAIRING_REQUEST_REPLY	0xFCAD	-	Y	-	Y	-
ACI_GAP_ADDITIONAL_BEACON_START	0xFCB0	-	Y	-	Y	-
ACI_GAP_ADDITIONAL_BEACON_STOP	0xFCB1	-	Y	-	Y	-
ACI_GAP_ADDITIONAL_BEACON_SET_DATA	0xFCB2	-	Y	-	Y	-
ACI_GAP_ADV_SET_CONFIGURATION	0xFCC0	-	-	-	-	-
ACI_GAP_ADV_SET_ENABLE	0xFCC1	-	-	-	-	-
ACI_GAP_ADV_SET_ADV_DATA	0xFCC2	-	-	-	-	-
ACI_GAP_ADV_SET_SCAN_RESP_DATA	0xFCC3	-	-	-	-	-
ACI_GAP_ADV_REMOVE_SET	0xFCC4	-	-	-	-	-
ACI_GAP_ADV_CLEAR_SETS	0xFCC5	-	-	-	-	-
ACI_GAP_ADV_SET_RANDOM_ADDRESS	0xFCC6	-	-	-	-	-
ACI_GAP_EXT_START_SCAN	0xFCD0	-	-	-	-	-
ACI_GAP_EXT_CREATE_CONNECTION	0xFCD1	-	-	-	-	-

2.4.1 ACI_GAP_SET_NON_DISCOVERABLE

Puts the device in non-discoverable mode. This command, which supports only legacy advertising, disables the LL advertising.

Input parameters: none

Table 170. 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

Puts the device in limited discoverable mode, making it discoverable for a maximum period of TGAP (lim_adv_timeout) = 180 seconds (from errata). The advertising can be disabled at any time by issuing 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 default values for adv intervals for limited discoverable mode (250 and 500 ms, respectively). To allow a fast connection, the host can set Local_Name, Service_Uuid_List, Conn_Interval_Min and Conn_Interval_Max. If provided, these data are inserted into the advertising packet payload as AD data: these parameters are optional, their values can be set in advertised data using the ACI_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 an ACI_GAP_LIMITED_DISCOVERABLE_EVENT

Table 171. **ACI_GAP_SET_LIMITED_DISCOVERABLE input parameters**

Parameter	Size	Description	Possible values
Advertising_Type	1	Advertising type	• 0x00: ADV_IND (connectable undirected 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 ms) ... 0x4000 (10240 ms)
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 ms	0x0020 (20 ms) ... 0x4000 (10240 ms)
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable or a non-resolvable private address.	• 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, 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 advertise, set this field to 0x00.	-
Service_Uuid_List	Service_Uuid_length	This is the list of the uuids. First byte is the AD Type.	-
Conn_Interval_Min	2	Peripheral connection interval minimum value suggested by peripheral. If Conn_Interval_Min and Conn_Interval_Max are not 0x0000, peripheral connection interval range AD structure is added in advertising data. Connection interval is defined as connIntervalmin = Conn_Interval_Min x 1.25 ms.	• 0x0000 (NaN) • 0xFFFF (NaN): no specific minimum • 0x0006 (7.50 ms) ... 0x0C80 (4000 ms)
Conn_Interval_Max	2	Peripheral connection interval maximum value suggested by peripheral. If Conn_Interval_Min and Conn_Interval_Max are not 0x0000, peripheral connection interval range AD structure is added in advertising data. Connection interval is defined as connIntervalmax = Conn_Interval_Max x 1.25 ms	• 0x0000 (NaN) • 0xFFFF (NaN) : no specific maximum • 0x0006 (7.50 ms) ... 0x0C80 (4000 ms)

Table 172. **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

This command, which supports only legacy advertising, puts 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 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 peripheral connection interval range. If provided, these data are inserted into the advertising packet payload as AD data. These parameters are optional in this command, their 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

Table 173. **ACI_GAP_SET_DISCOVERABLE input parameters**

Parameter	Size	Description	Possible values
Advertising_Type	1	Advertising type.	• 0x00: ADV_IND (connectable undirected 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 ms) ... 0x4000 (10240 ms)
Advertising_Interval_Max	2	Maximum advertising interval Time = N * 0.625 ms	0x0020 (20 ms) ... 0x4000 (10240 ms)
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable or a non-resolvable private address.	• 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)	-

Parameter	Size	Description	Possible values
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.	-
Conn_Interval_Min	2	Peripheral connection interval minimum value suggested by peripheral. If Conn_Interval_Min and Conn_Interval_Max are not 0x0000, peripheral connection interval range AD structure is added in advertising data. Connection interval is defined as: connIntervalmin = Conn_Interval_Min x 1.25 ms.	• 0x0000 (NaN) • 0xFFFF (NaN): no specific minimum • 0x0006 (7.50 ms) ... 0x0C80 (4000 ms)
Conn_Interval_Max	2	Peripheral connection interval maximum value suggested by peripheral. If Conn_Interval_Min and Conn_Interval_Max are not 0x0000, peripheral connection interval range AD structure is added in advertising data. Connection interval is defined as: connIntervalmax = Conn_Interval_Max x 1.25 ms.	• 0x0000 (NaN) • 0xFFFF (NaN): no specific maximum • 0x0006 (7.50 ms) ... 0x0C80 (4000 ms)

Table 174. 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

This command supports only legacy advertising, and sets the device in direct connectable mode.

The device advertises using high or low duty cycle connectable directed advertising events. The address used in advertising packets is defined by the Own_Address_Type parameter, depending upon privacy (enabled or not).

When using high duty cycle connectable directed advertising events, the device stays in directed connectable mode only for 1.28 seconds. If no connection is established within this duration, it enters non discoverable mode, and advertising must be re-enabled explicitly. The controller generates HCI_LE_CONNECTION_COMPLETE_EVENT with the status set to HCI_ADVERTISING_TIMEOUT_ERR_CODE if the connection is not established, BLE_STATUS_SUCCESS (0x00) if the connection is successfully established.

Table 175. ACI_GAP_SET_DIRECT_CONNECTABLE input parameters

Parameter	Size	Description	Possible values
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable private address.	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address

Parameter	Size	Description	Possible values
Directed_Advertising_Type	1	Advertising type	• 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 (3.75 ms): for high duty cycle directed advertising • 0x0020 (20 ms) ... 0x4000 (10240 ms): for low duty cycle directed advertising
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 msec.	• 0x0006 (3.75 ms) : for high duty cycle directed advertising • 0x0020 (20 ms) ... 0x4000 (10240 ms) : for low duty cycle directed advertising

Table 176. 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

This command set the device IO capabilities. It must be given only when the device is not in a connected state.

Table 177. 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

Table 178. 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

Sets 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 must be given only when the device is not in a connected state.

Table 179. **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 use during pairing	-
Use_Fixed_Pin	1	Use or not fixed pin. If set to 0x00, during the pairing process the application is not requested for a pin (Fixed_Pin is used). If set to 0x01, 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 use during pairing if MIMT protection is enabled (random value)	0 ... 999999
Identity_Address_Type	1	Identity address type	• 0x00: Public identity address • 0x01: Random (static) identity address

Table 180. **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

Sets 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.

Table 181. **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

Table 182. **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

This command is sent by the host in response to an ACI_GAP_PASS_KEY_REQ_EVENT. The command parameters contain the pass key used during the pairing process.

Table 183. **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 used during the pairing process. Must be a six-digit decimal number.	0 ... 999999

Table 184. **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

This command is sent by the host in response to an ACI_GAP_AUTHORIZATION_REQ_EVENT, to authorize a device to access attributes.

Table 185. **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

Table 186. **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

Initializes the GAP layer, and registers the GAP service with the GATT. All the standard GAP characteristics are added:

- device name
- appearance
- peripheral preferred connection parameters (peripheral role only): if added, its handle is equal to the Appearance characteristic handle plus 2

If privacy is enabled, the command unmasks the HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT.

Table 187. 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	Specifies if privacy is enabled or not (only Controller privacy is supported)	• 0x00: Privacy disabled • 0x02: Privacy enabled
device_name_char_len	1	Length of the device name characteristic	-

Table 188. 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

Puts the device into non connectable mode, not supporting connections. 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.

Table 189. 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: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable or a non resolvable private address.	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address • 0x03: Non resolvable private address

Table 190. **ACI_GAP_SET_NON_CONNECTABLE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status Error code	-

- HCI_COMMAND_COMPLETE_EVENT

2.4.12 ACI_GAP_SET_UNDIRECTED_CONNECTABLE

This command, which supports only legacy advertising, puts the device into undirected connectable mode. If privacy is enabled, 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.

Table 191. **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 ms) ... 0x4000 (10240 ms)
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 ms	0x0020 (20 ms) ... 0x4000 (10240 ms)
Own_Address_Type	1	Own address type: if privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable private address.	• 0x00: Public device address • 0x01: Random device address • 0x02: Resolvable private address
Adv_Filter_Policy	1	Advertising filter policy	• 0x00: Allow scan and connect requests from any • 0x03: Allow scan and connect requests only from white list

Table 192. **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_PERIPHERAL_SECURITY_REQ

This command sends a security request to the central, to notify its security requirements. The central can encrypt the link, initiate the pairing procedure, or reject the request.

Table 193. **ACI_GAP_PERIPHERAL_SECURITY_REQ input parameters**

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

Table 194. **ACI_GAP_PERIPHERAL_SECURITY_REQ output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT

- ACI_GAP_PERIPHERAL_SECURITY_INITIATED_EVENT

2.4.14 ACI_GAP_UPDATE_ADV_DATA

This command can be used dinamically (when advertising is already enabled) to update the data for a particular AD type: if the specified type does not exist, it is added. If the overall advertising data length is more than 31 octets after the update the command is rejected and the old data are retained.

Table 195. 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 196. 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

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

Table 197. ACI_GAP_DELETE_AD_TYPE input parameters

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

Table 198. 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

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

Table 199. 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

Table 200. **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

This command makes it possible to mask events from the GAP. If the bit in the GAP_Evt_Mask is set to 1, the event associated with that bit is enabled. The default configuration is all the events masked.

Table 201. **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_PERIPHERAL_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

Table 202. **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_FILTER_ACCEPT_LIST

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

Input parameters: none

Table 203. **ACI_GAP_CONFIGURE_FILTER_ACCEPT_LIST output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.19 ACI_GAP_TERMINATE

Command the controller to terminate the connection. An HCI_DISCONNECTION_COMPLETE_EVENT is generated when the link 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, the system saves important information in the non volatile memory.

Table 204. 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

Table 205. 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

Clears the bonding table, all the devices are removed from it. See ACI_GAP_REMOVE_BONDED_DEVICE to remove only one device.

Note: As a fallback mode, if the bonding table is full, the BLE stack automatically clears it before writing information about a new bonded device.

Input parameters: none

Table 206. 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

Allows the security manager to complete the pairing procedure and re-bond with the central. 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 times out.

Table 207. **ACI_GAP_ALLOW_REBOND input parameters**

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

Table 208. **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

Starts 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 the command ACI_GAP_TERMINATE_GAP_PROC with the procedure code set to 0x01, or there is a timeout (value fixed at 10.24 s). When the procedure is terminated, ACI_GAP_PROC_COMPLETE_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 HCI_LE_ADVERTISING_REPORT_EVENT (or via HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT when the extended advertising feature is supported).

Table 209. **ACI_GAP_START_LIMITED_DISCOVERY_PROC input parameters**

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	The time interval from when the controller started its last LE scan, until it begins the subsequent one. Time = N * 0.625 ms	0x0004 (2.50 ms) ... 0x4000 (10240 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 (2.50 ms) ... 0x4000 (10240 ms)
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable or a non-resolvable private address.	• 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 210. **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
- HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT

- ACI_GAP_PROC_COMPLETE_EVENT

2.4.23 ACI_GAP_START_GENERAL_DISCOVERY_PROC

Starts the general discovery procedure. The controller is commanded to start active scanning. The procedure is terminated when either the upper layers issue the command ACI_GAP_TERMINATE_GAP_PROC with the procedure code set to 0x02, or when a timeout happens (value fixed at 10.24 s). When the procedure is terminated, ACI_GAP_PROC_COMPLETE_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 (or via HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT when the extended advertising feature is supported).

Table 211. ACI_GAP_START_GENERAL_DISCOVERY_PROC input parameters

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	Time interval from when the controller started its last LE scan, until it begins the subsequent one. Time = N * 0.625 ms	• 0x0004 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
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 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable or a non-resolvable private address.	• 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

Table 212. 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
- HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.4.24 ACI_GAP_START_AUTO_CONNECTION_ESTABLISH_PROC

Starts the auto connection establishment procedure. The specified devices are added to the white list of the controller, and an 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 when it is explicitly terminated by issuing the command ACI_GAP_TERMINATE_GAP_PROC with the procedure code set to 0x08. An ACI_GAP_PROC_COMPLETE_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, the link layer generates an RPA, if not, the RPA/NRPA generated by the host is used.

Table 213. ACI_GAP_START_AUTO_CONNECTION_ESTABLISH_PROC input parameters

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	Time interval from when the controller started its last LE scan until it begins the subsequent one. Time = N * 0.625 ms	• 0x0004 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
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 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable or a non-resolvable private address.	• 0x00: Public address • 0x01: Static random 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 (7.5 ms) ... 0x0C80 (4000 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 (7.5 ms) ... 0x0C80 (4000.00 ms)
Conn_Latency	2	Peripheral 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 + connPeripheralLatency) * 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 address • 0x01: Random address
Peer_Address[i]	6	Public or random address of the device to be added to the white list.	-

Table 214. 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

2.4.25 ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC

Starts a general connection establishment procedure. The host enables scanning in the controller with policy set to "accept all advertising packets" and from the scanning results, all the devices are sent to the upper layer using HCI_LE_ADVERTISING_REPORT_EVENT (or by HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT when the extended advertising feature is supported). The upper layer then selects the device to whom it wants to connect, by issuing the command ACI_GAP_CREATE_CONNECTION. If privacy is enabled, a private resolvable or non resolvable address (based on the type specified in the command) is set as scanner address. The ACI_GAP_CREATE_CONNECTION command 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, an ACI_GAP_PROC_COMPLETE_EVENT is generated with code set to 0x10. If controller privacy is enabled and the peer device (advertiser) is in the resolving list, the link layer generates an RPA, if not, the RPA/NRPA generated by the host is used.

Table 215. 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	Time interval from when the controller started its last LE scan, until it begins the subsequent one. Time = N * 0.625 ms	• 0x0004 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
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 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable or a non-resolvable private address.	• 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 packets not addressed to this device are ignored. • 0x01 Ignore advertisement packets from devices not in the White list. Directed packets not addressed to this device are ignored. • 0x02 Accept all undirected advertisement packets (allowed only if controller privacy is enabled). Directed packets where initiator address is an RPA and directed packets addressed to this device accepted. • 0x03 Accept all undirected advertisement packets from devices in the White list. Directed packets where initiator address is an RPA and directed packets addressed to this device are accepted. If controller privacy is enabled Scanning_Filter_Policy can be only 0x00 or 0x02.	• 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

Table 216. **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
- HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.4.26 ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC

Starts a selective connection establishment procedure. The GAP adds the specified device addresses in the white list, and enables scanning in the controller with the filter policy set to "accept packets only from devices in white list". All found devices are sent to the upper layer by the HCI_LE_ADVERTISING_REPORT_EVENT (or by HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT when the extended advertising feature is supported). The upper layer then selects the device to whom it wants to connect, by issuing the command ACI_GAP_CREATE_CONNECTION. Before the call, the procedure can be terminated by issuing the command ACI_GAP_TERMINATE_GAP_PROC with the procedure code set to 0x20. After the call, 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 0x40. On completion, ACI_GAP_PROC_COMPLETE_EVENT is generated with the procedure code set to 0x40.

If privacy is enabled and the peer device (advertiser) is in the resolving list, the link layer generates an RPA.

Table 217. **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	Time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	• 0x0004 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
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 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable or a non-resolvable private address.	• 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 not addressed to this device are ignored. • 0x01 Ignore advertisement packets from devices not in the white list. Directed advertising packets not addressed to this device are ignored.	• 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
		• 0x02 Accept all undirected advertisement packets (allowed only if controller privacy is enabled). Directed advertisement packets where initiator address is an RPA and directed advertisement packets addressed to this device are accepted. • 0x03 Accept all undirected advertisement packets from devices in the white list. Directed advertisement packets where initiator address is RPA and directed advertisement packets addressed to this device are accepted. *Note: If controller privacy is enabled the value can be only 0x01 or 0x03.*	
Filter_Duplicates	1	Enable/disable duplicate filtering	• 0x00: Duplicate filtering disabled • 0x01: Duplicate filtering enabled
Num_of_Whitelist_Entries	1	Number of devices to add to the white list	-
Peer_Address_Type[i]	1	Address type	• 0x00: Public device address • 0x01: Random device address
Peer_Address[i]	6	Public or random address of the device to add to the white list.	-

Table 218. 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
- HCI_LE_ADVERTISING_REPORT_EVENT
- HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT

2.4.27 ACI_GAP_CREATE_CONNECTION

Starts the direct connection establishment procedure.

A HCI_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 this happens, a HCI_LE_CREATE_CONNECTION_CANCEL call is made to the controller. On termination of the procedure, a HCI_LE_CONNECTION_COMPLETE_EVENT (or HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT if privacy or extended advertising is used) is returned. The procedure can be explicitly terminated by the upper layer by issuing the command ACI_GAP_TERMINATE_GAP_PROC with procedure_code set to 0x40.

If controller privacy is enabled and the peer device (advertiser) is in the resolving list, the link layer generates an RPA. If not, the RPA/NRPA generated by the host is used.

Table 219. ACI_GAP_CREATE_CONNECTION input parameters

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	Time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	• 0x0004 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
LE_Scan_Window	2	Amount of time for the duration of the LE scan. LE_Scan_Window must less than or equal to LE_Scan_Interval. Time = N * 0.625 ms	• 0x0004 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
Peer_Address_Type	1	The address type of the peer device.	• 0x00: Public address • 0x01: Random address
Peer_Address	6	Public device address or random device address of the device to be connected.	-
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable private address.	• 0x00: Public address • 0x01: Random address • 0x02: 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 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 ms)
Conn_Latency	2	Peripheral latency for the connection, in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout	2	Supervision timeout for the LE link. Must be a multiple of 10 ms, and larger than (1 + connPeripheralLatency) * 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)

Table 220. 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
- HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT

2.4.28 ACI_GAP_TERMINATE_GAP_PROC

Terminates the specified GATT procedure. An ACI_GAP_PROC_COMPLETE_EVENT is returned, with the procedure code set to the corresponding procedure.

Table 221. 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

Table 222. 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

2.4.29 ACI_GAP_START_CONNECTION_UPDATE

Starts the connection update procedure (only when role is ,central). An HCI_LE_CONNECTION_UPDATE is called. On completion of the procedure, an HCI_LE_CONNECTION_UPDATE_COMPLETE_EVENT is returned to the upper layer.

Table 223. 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. 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	Peripheral 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 + connPeripheralLatency) * 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)

Table 224. **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

2.4.30 ACI_GAP_SEND_PAIRING_REQ

Sends 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. An ACI_GAP_PAIRING_COMPLETE_EVENT returned after the pairing process is completed.

Table 225. **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

Table 226. **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

2.4.31 ACI_GAP_RESOLVE_PRIVATE_ADDR

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.

Table 227. **ACI_GAP_RESOLVE_PRIVATE_ADDR input parameters**

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

Output parameters

Table 228. **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

2.4.32 ACI_GAP_SET_BROADCAST_MODE

This command, which supports only legacy advertising, 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.

Table 229. 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 ms) ... 0x4000 (10240 ms)
Advertising_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 ms	0x0020 (20ms) ... 0x4000 (10240 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 the address can be public or static random. Otherwise, it can be a resolvable 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	-
Adv_Data	Adv_Data_Length	Advertising data used by the device while advertising	-
Num_of_Whitelist_Entries	1	Number of devices to add 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	-

Table 230. ACI_GAP_SET_BROADCAST_MODE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.33 ACI_GAP_START_OBSERVATION_PROC

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, the link layer generates an RPA, if not, the RPA/NRPA generated by the host is used.

Table 231. ACI_GAP_START_OBSERVATION_PROC input parameters

Parameter	Size	Description	Possible values
LE_Scan_Interval	2	Time interval from when the controller started its last LE scan until it begins the subsequent LE scan. Time = N * 0.625 ms	• 0x0004 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
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 (2.5 ms) ... 0x4000 (10240 ms): legacy advertsing • 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): extended advertsing
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: if Privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable or a non-resolvable private address.	• 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 not addressed to this device are ignored. • 0x01 Ignore advertisement packets from devices not in the white list. Only Directed advertising packets not addressed to this device are ignored. • 0x02 Accept all undirected advertisement packets (it is allowed only if controller privacy is enabled). Directed advertisement packets where initiator address is a RPA and directed advertisement packets addressed to this device are 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 are 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)

Table 232. 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
- HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT

2.4.34 ACI_GAP_GET_BONDED_DEVICES

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

Table 233. 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 add to the white list	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.35 ACI_GAP_IS_DEVICE_BONDED

Finds whether the device, whose address is specified in the command, is present in the bonding table. If the device is found, the command returns BLE_STATUS_SUCCESS.

The specified address can be an RPA. In this case, even if privacy is not enabled, this address is resolved to check the presence of the peer device in the bonding table.

Table 234. ACI_GAP_IS_DEVICE_BONDED input parameters

Parameter	Size	Description	Possible values
Peer_Address_Type	1	Address type of the peer device	• 0x00: Public device address • 0x01: Random device address
Peer_Address	6	Public or random device address of the peer device	-

Table 235. ACI_GAP_IS_DEVICE_BONDED output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.36 ACI_GAP_NUMERIC_COMPARISON_VALUE_CONFIRM_YESNO

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

Table 236. 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

Table 237. **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

2.4.37 ACI_GAP_PASSKEY_INPUT

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

Table 238. **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

Table 239. **ACI_GAP_PASSKEY_INPUT output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.38 ACI_GAP_GET_OOB_DATA

This command is sent by the server to get (extract) the OOB data generated by the stack.

Table 240. **ACI_GAP_GET_OOB_DATA input parameters**

Parameter	Size	Description	Possible values
OOB_Data_Type	1	OOB data type	• 0x00: TK (legacy pairing) • 0x01: Random (SC) • 0x02: Confirm (SC)

Table 241. **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 (legacy pairing) • 0x01: Random (SC) • 0x02: Confirm (SC)
OOB_Data_Len	1	Length of OOB data	16
OOB_Data	16	Local OOB data	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.39 ACI_GAP_SET_OOB_DATA

This command is sent by the user to input the OOB data received via OOB communication.

Table 242. ACI_GAP_SET_OOB_DATA input parameters

Parameter	Size	Description	Possible values
Device_Type	1	OOB device type	• 0x00: Local device (Address_Type and Address are not used) • 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 (legacy pairing) • 0x01: Random (SC) • 0x02: Confirm (SC)
OOB_Data_Len	1	Length of OOB data	• 0: SC Random/Confirm generation (OOB_Data and OOB_Data_Type are not used) • 1:
OOB_Data	16	Local or remote OOB data received through OOB from peer device (see Device_Type)	-

Table 243. ACI_GAP_SET_OOB_DATA output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.40 ACI_GAP_ADD_DEVICES_TO_RESOLVING_LIST

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

Table 244. 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	• 0x00: Do not clear • 0x01: Clear before adding

Table 245. **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

2.4.41 ACI_GAP_REMOVE_BONDED_DEVICE

This command is used to remove a specified device from the bonding table. It returns the number of addresses and the corresponding address types and values.

Table 246. **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	-

Table 247. **ACI_GAP_REMOVE_BONDED_DEVICE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.42 ACI_GAP_ADD_DEVICES_TO_LIST

This command is used by the Host to add specific device addresses to the white and/or resolving lists.

Table 248. **ACI_GAP_ADD_DEVICES_TO_LIST input parameters**

Parameter	Size	Description	Possible values
Num_of_Resolving_list_Entries	1	Number of devices to add to the list	-
Address_Type[i]	1	Address type	• 0x00: public device address • 0x01: random device address
Address[i]	6		-
Mode	1		• 0x00: append only to the resolving list • 0x01: clear and set only the resolving list • 0x02: append only to the white list • 0x03: clear and set only the white list • 0x04: append to both lists • 0x05: clear and set both lists

Table 249. **ACI_GAP_ADD_DEVICES_TO_LIST output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.43 ACI_GAP_PAIRING_REQUEST_REPLY

This command is used to reply to ACI_GAP_PAIRING_REQUEST_EVENT. It makes possible to accept or reject the pairing request from the central or the security request from the peripheral.

Table 250. ACI_GAP_PAIRING_REQUEST_REPLY input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle to whom the command applies	0x0000 … 0x0EFF
Accept	1	Enables acceptance or rejection of the pairing request	• 0x00: Reject • 0x01: Accept

Table 251. ACI_GAP_PAIRING_REQUEST_REPLY output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.44 ACI_GAP_ADDITIONAL_BEACON_START

This command starts an advertising beacon: additional advertising packets can be transmitted independently from the packets transmitted with GAP advertising commands, such as ACI_GAP_SET_DISCOVERABLE or ACI_GAP_SET_LIMITED_DISCOVERABLE.

Table 252. ACI_GAP_ADDITIONAL_BEACON_START input parameters

Parameter	Size	Description	Possible values
Adv_Interval_Min	2	Minimum advertising interval. Time = N * 0.625 msec.	• 0x0020: 20 ms • 0x4000: 10240 ms
Adv_Interval_Max	2	Maximum advertising interval. Time = N * 0.625 msec.	• 0x0020: 20 ms • 0x4000: 10240 ms
Adv_Channel_Map	1	Advertising channel map. Default: 00000111b (all channels enabled).	Bitmask of • 0x01: use Channel 37 • 0x02: use Channel 38 • 0x04: use Channel 39
Own_Address_Type	1	Own address type: public or static random	• 0x00: Public device address • 0x01: Random device address
Own_Address	6	Public or random device address	-
PA_Level	1	Power amplifier output level. Output power is indicative and depends on the PCB layout and associated components. The inaccuracy can exceed 10 dB in the range -20.85 to -40 dBm. Here the values are given at the STM32WB output.	• 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

Table 253. ACI_GAP_ADDITIONAL_BEACON_START output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.45 ACI_GAP_ADDITIONAL_BEACON_STOP

This command stops the advertising beacon started with ACI_GAP_ADDITIONAL_BEACON_START.

Input parameters: none

Table 254. ACI_GAP_ADDITIONAL_BEACON_STOP output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.46 ACI_GAP_ADDITIONAL_BEACON_SET_DATA

This command sets the data transmitted by the advertising beacon started with ACI_GAP_ADDITIONAL_BEACON_START. If the advertising beacon is already started, the new data are used in subsequent beacon advertising events.

Table 255. ACI_GAP_ADDITIONAL_BEACON_SET_DATA input parameters

Parameter	Size	Description	Possible values
Adv_Data_Length	1	Length of Adv_Data in octets	-
Adv_Data	Adv_Data_Length	Advertising data used by the device while advertising	-

Table 256. ACI_GAP_ADDITIONAL_BEACON_SET_DATA output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.47 ACI_GAP_ADV_SET_CONFIGURATION

This command is used to set the extended advertising configration for one advertising set. In association with ACI_GAP_ADV_SET_SCAN_RESP_DATA, ACI_GAP_ADV_SET_ADV_DATA, and ACI_GAP_ADV_SET_ENABLE, it enables to start extended advertising.

These commands must be used to replace ACI_GAP_SET_DISCOVERABLE, ACI_GAP_SET_LIMITED_DISCOVERABLE, ACI_GAP_SET_DIRECT_CONNECTABLE, ACI_GAP_SET_NON_CONNECTABLE, ACI_GAP_SET_UNDIRECTED_CONNECTABLE, and ACI_GAP_SET_BROADCAST_MODE, supporting only legacy advertising.

If bit 0 of Adv_Mode is set, the Own_Address_Type parameter is ignored and Own_Address is set with the ACI_GAP_ADV_SET_RANDOM_ADDRESS command. This mode is valid only for non-connectable advertising.

If bit 1 of Adv_Mode is set, the primary advertisement PHY is set to LE coded (not supported on STM32WB), otherwise, the default primary advertisement PHY is LE 1M.

To configure a periodic advertising set, use the command ACI_GAP_ADV_SET_PERIODIC_PARAMETERS (in association with ACI_GAP_ADV_SET_PERIODIC_DATA and ACI_GAP_ADV_SET_PERIODIC_ENABLE) instead of this one (feature not supported on STM32WB).

Table 257. ACI_GAP_ADV_SET_CONFIGURATION input parameters

Parameter	Size	Description	Possible values
Adv_Mode	1	Bitmap of extended advertising modes	Bitmask of: • 0x01: use specific random address • 0x02: use LE coded as primary advertising PHY (not supported by STM32WB devices)

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 ... 0xEF
Adv_Event_Properties	2	Type of advertising event	Bitmask of: • 0x0001: connectable advertising • 0x0002: scannable advertising • 0x0004: directed advertising • 0x0008: high duty cycle directed connectable advertising • 0x0010: use legacy advertising PDUs • 0x0020: anonymous advertising • 0x0040: include TxPower in at least one advertising PDU
Primary_Adv_Interval_Min	4	Minimum advertising interval. Time = N * 0.625 ms.	0x00000020 (20.0 ms) ... 0x00FFFFFF (10485759.375 ms)
Primary_Adv_Interval_Max	4	Maximum advertising interval. Time = N * 0.625 ms.	0x00000020 (20.0 ms) ... 0x00FFFFFF (10485759.375 ms)
Primary_Adv_Channel_Map	1	Advertising channel map.	Bitmask of: • 0x01: use channel 37 • 0x02: use channel 38 • 0x04: use channel 39
Own_Address_Type	1	Own address type: if Privacy is disabled, it can be public or static random, otherwise, it can be a resolvable or a non-resolvable private address.	• 0x00: public address • 0x01: static random address • 0x02: resolvable private address • 0x03: non-resolvable private address
Peer_Address_Type	1	Address type of the peer device	• 0x00: public device or public identity address • 0x01: static random address • 0x02: random device or random (static) identity address
Peer_Address	6	Public or random device, public or random (static) identity address of the device to connect	-
Adv_Filter_Policy	1	Advertising filter policy	• 0x00: process scan and connection requests from all devices (white list not in use) • 0x01: process connection requests from all devices and scan requests only from devices in the white list • 0x02: process scan requests from all devices and connection requests only from devices in the white list • 0x03: process scan and connection requests only from devices in the white list
Adv_TX_Power	1	Advertising TX power. Units: dBm.	• 127: host has no preferences • -127 ... 20
Secondary_Adv_Max_Skip	1	Secondary advertising maximum skip	• 0x00: AUX_ADV_IND must be sent prior to the next advertising event • 0x01 ... 0xFF: maximum advertising events that the controller can skip before sending the AUX_ADV_IND packets on the secondary advertising physical channel
Secondary_Adv_PHY	1	Secondary advertising PHY	• 0x01: secondary advertisement PHY is LE 1M • 0x02: secondary advertisement PHY is LE 2M • 0x03: secondary advertisement PHY is LE coded (not supported by STM32WB devices)
Adv_SID	1	Value of the advertising SID subfield in the ADI field of PDU	0x00 ... 0x0F
Scan_Req_Notification_Enable	1	Scan request notifications	• 0x00: scan disabled • 0x01: scan enabled

Table 258. **ACI_GAP_ADV_SET_CONFIGURATION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.48 ACI_GAP_ADV_SET_ENABLE

This command is used to request the controller to enable/disable one or more extended advertising sets.

Table 259. **ACI_GAP_ADV_SET_ENABLE input parameters**

Parameter	Size	Description	Possible values
Enable	1	Enable/disable advertising	• 0x00: advertising disabled • 0x01: advertising enabled
Num_Sets	1	Number of advertising sets	• 0x00: disable all advertising sets • 0x01 ... 0x3F: number of advertising sets to enable/disable
Advertising_Handle[i]	1	Used to identify an advertising set	0x00 … 0xEF
Duration[i]	2	Duration of advertising set. Time = N * 10 ms.	• 0x0000 (0 ms): no advertising duration • 0x0001 (10 ms) … 0xFFFF (655350 ms): advertising duration
Max_Extended_Advertising_Events[i]	1	Maximum number of advertising events	• 0x00: no maximum number of advertising events • 0x01 ... 0xFF: maximum number of extended advertising events the controller attempts to send prior to terminate the extended advertising

Table 260. **ACI_GAP_ADV_SET_ENABLE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.49 ACI_GAP_ADV_SET_ADV_DATA

This command is used to request the controller to enable/disable one or more extended advertising sets.

Table 261. **ACI_GAP_ADV_SET_ADV_DATA input parameters**

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 … 0xEF
Operation	1	Advertising operation	• 0x00: Intermediate fragment of fragmented extended advertising data • 0x01: first fragment of fragmented extended advertising data • 0x02: last fragment of fragmented extended advertising data

Parameter	Size	Description	Possible values
			• 0x03: complete extended advertising data • 0x04: unchanged data (just update the advertising DID)
Fragment_Preference	1	Fragment preference	• 0x00: the controller can fragment data • 0x01: the controller must not fragment data, or minimize fragmentation
Advertising_Data_Length	1	Length of Advertising_Data in octets	-
Advertising_Data	Advertising_Data_Length	Data formatted as defined in Bluetooth spec. v.5.2 [Vol 3, Part C, 11].	-

Table 262. ACI_GAP_ADV_SET_ADV_DATA output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.50 ACI_GAP_ADV_SET_SCAN_RESP_DATA

This command is used to provide scan response data used during extended advertising.

Table 263. ACI_GAP_ADV_SET_SCAN_RESP_DATA input parameters

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 … 0xEF
Operation	1	Advertising operation	• 0x00: intermediate fragment of fragmented extended advertising data • 0x01: first fragment of fragmented extended advertising data • 0x02: last fragment of fragmented extended advertising data • 0x03: complete scan response data
Fragment_Preference	1	Fragment preference	• 0x00: the controller may fragment all data • 0x01: the controller must not fragment data, or minimize fragmentation
Scan_Response_Data_Length	1	Length of Scan_Response_Data in octets	-
Scan_Response_Data	Scan_Response_Data_Length	Data formatted as defined in Bluetooth spec. v.5.2 [Vol 3, Part C, 11].	-

Table 264. **ACI_GAP_ADV_SET_SCAN_RESP_DATA output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.51 ACI_GAP_ADV_REMOVE_SET

This command is used to request the controller to enable/disable one or more extended advertising sets.

Table 265. **ACI_GAP_ADV_REMOVE_SET input parameters**

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 … 0xEF

Table 266. **ACI_GAP_ADV_REMOVE_SET output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.52 ACI_GAP_ADV_CLEAR_SETS

This command is used to remove all existing advertising sets from the controller.

Input parameters: none

Table 267. **ACI_GAP_ADV_CLEAR_SETS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.53 ACI_GAP_ADV_SET_RANDOM_ADDRESS

This command is used to set the address of an advertising set configured to use specific random address.

Table 268. **ACI_GAP_ADV_SET_RANDOM_ADDRESS input parameters**

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 … 0xEF
Random_Address	6	Random device address	-

Table 269. **ACI_GAP_ADV_SET_RANDOM_ADDRESS output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.4.54 ACI_GAP_EXT_START_SCAN

This command is used to start a scanning procedure when the extended advertising feature is supported. It can be used instead of ACI_GAP_START_LIMITED_DISCOVERY_PROC, ACI_GAP_START_GENERAL_DISCOVERY_PROC, ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC, ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC, or ACI_GAP_START_OBSERVATION_PROC, as it is more generic.

The Scanning_PHYs and subsequent parameters are used to specify the scanning parameters as defined for HCI_LE_SET_EXTENDED_SCAN_PARAMETERS. For more details, refer to Bluetooth spec. v.6.0 [Vol 4, Part E, 7.8.64].

Table 270. ACI_GAP_EXT_START_SCAN input parameters

Parameter	Size	Description	Possible values
Scan Mode	1	Not used, set it to 0	-
Procedure	1	Scan procedure	• 0x01: GAP_LIMITED_DISCOVERY_PROC • 0x02: GAP_GENERAL_DISCOVERY_PROC • 0x10: GAP_GENERAL_CONNECTION_ESTABLISHMENT_PROC • 0x20: GAP_SELECTIVE_CONNECTION_ESTABLISHMENT_PROC • 0x80: GAP_OBSERVATION_PROC
Own_Address_Type	1	Own address type: if Privacy is disabled, the address can be public or static random; otherwise, it can be a resolvable or a non-resolvable private address.	• 0x00: Public address • 0x01: Static random address • 0x02: Resolvable private address • 0x03: Non-resolvable private address
Filter_Duplicate	1	Enable/disable duplicate filtering	• 0x00: Duplicate filtering disabled • 0x01: Duplicate filtering enabled
Duration	2	Scan duration. Time = N * 10 ms.	• 0x000 (0 ms) : Scan continuously until explicitly disabled • 0x001 (10 ms) ... 0xFFFF (655350 ms): Scan duration
Period	2	Scan period. Time = N * 1.28 s.	• 0x0000 (0 ms): Scan continuously • 0x0001 (1280 ms) ... 0xFFFF (83884800 ms): Time interval from when the controller started its last Scan_Duration until it begins the subsequent one
Scanning_Filter_Policy	1	Determines how the scanner's Link Layer processes advertising and scan response PDUs. There are two primary filter policies: • Unfiltered: the Link Layer processes all advertising and scan response PDUs (the Filter Accept List is not used)	• 0x00: Basic unfiltered scanning filterpolicy • 0x01: Basic filtered scanning filterpolicy • 0x02: Extended unfiltered scanning filter policy • 0x03: Extended filtered scanning filter policy

Parameter	Size	Description	Possible values
		• Filtered: the Link Layer processes advertising and scan response PDUs only from devices in the Filter Accept List. With extended scanning filter policies, a directed advertising PDU accepted by the primary filter policy must be ignored unless the TargetA field is identical to the scanner's device address, or TargetA field is a resolvable private address.	
Scanning_PHYs	1	Scan PHYs	Bitmask of: • 0x01: Scan advertisements on LE 1M PHY • 0x04: Scan advertisements on LE coded PHY (not supported by STM32WB)
Scan_Type[i]	1	Passive or active scanning. With passive scanning, no scan request PDUs are sent	• 0x00: Passive scanning • 0x01: Active scanning
Scan_Interval[i]	1	Time interval from when the controller started its last scan until it begins the subsequent one on the primary advertising physical channel. Time = N * 0.625 ms.	• 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): STM32WB • 0x0004 (2.5 ms) ... 0xFFFF (40959.375 ms): STM32WBA
Scan_Window[i]	2	Scan duration on the primary advertising physical channel. Time = N * 0.625 ms.	• 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): STM32WB • 0x0004 (2.5 ms) ... 0xFFFF (40959.375 ms): STM32WBA

Table 271. ACI_GAP_EXT_START_SCAN output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.4.55 ACI_GAP_EXT_CREATE_CONNECTION

This command is used to create a connection with the local device in the Central role to an advertiser when the extended advertising feature is supported. This command can be used instead of ACI_GAP_CREATE_CONNECTION or ACI_GAP_START_AUTO_CONNECTION_ESTABLISH_PROC, as it is more generic.

The Advertising_Handle and Subevent parameters must be used for a connection to a periodic advertiser, otherwise set them to 0xFF. These parameters are ignored by STM32WB devices.

The Initiating_PHYs and subsequent parameters are used to specify the initiating parameters as defined for HCI_LE_EXTENDED_CREATE_CONNECTION: for more details, refer to Bluetooth spec. v.6.0 [Vol 4, Part E,7.8.66].

Table 272. ACI_GAP_EXT_CREATE_CONNECTION input parameters

Parameter	Size	Description	Possible values
Initiating_Mode	1	Not used, set it to 0	-
Procedure	1	Connection procedure	• 0x08: GAP_AUTO_CONNECTION_ESTABLISHMENT_PROC • 0x40: GAP_DIRECT_CONNECTION_ESTABLISHMENT_PROC
Own_Address_Type	1	Own address type: if privacy is disabled, the address can be public or static random, otherwise, it can be a resolvable private address.	• 0x00: Public address • 0x01: Static random address • 0x02: Resolvable private address
Peer_Address_Type	1	Address type of the peer device	• 0x00: Public address • 0x01: Random address
Peer_Address	6	Public or random device, public or random (static) identity address of the device to connect	-
Advertising_Handle	1	Used to identify the subevent where a connection request must be initiated from a periodic advertising train	• 0xFF: parameter not used • 0x00 … 0xEF
Subevent	1	Subevent where the connection request must be sent	• 0xFF: parameter not used • 0x00 … 0xEF
Initiator_Filter_Policy	1	Initiator filter policy	• 0x00: Filter accept list is not used to determine which advertiser to connect to • 0x01: Filter accept list is used to determine which advertiser to connect to (Peer_Address_Type and Peer_Address are ignored)
Initiating_PHYs	1	Initiating_PHYs	Bitmask of: • 0x01: Scan connectable advertisements on the LE 1M PHY • 0x02: Scan connectable advertisements on the LE 2M PHY • 0x04: Scan connectable advertisements on the LE coded PHY (not supported by STM32WB)
Scan_Interval[i]	2	Time interval from when the controller started its last scan until it begins the subsequent one on the primary advertising physical channel. Time = N * 0.625 ms.	• 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): STM32WB • 0x0004 (2.5 ms) ... 0xFFFF (40959.375 ms): STM32WBA

Parameter	Size	Description	Possible values
Scan_Window[i]	2	Duration of the scan on the primary advertising physical channel. Time = N * 0.625 ms.	• 0x0004 (2.5 ms) ... 0x5DC0 (15000 ms): STM32WB • 0x0004 (2.5 ms) ... 0xFFFF (40959.375 ms): STM32WBA
Conn_Interval_Min[i]	2	Minimum value for the connection event interval. Time = N * 1.25 ms.	0x0006 (7.5 ms) ... 0x0C80 (4000 ms)
Conn_Interval_Max[i]	2	Minimum value for the connection event interval. Time = N * 1.25 ms.	0x0006 (7.5 ms) ... 0x0C80 (4000 ms)
Conn_Latency[i]	2	Maximum peripheral latency for the connection in number of connection events.	0x0000 ... 0x01F3
Supervision_Timeout[i]	2	Supervision timeout for the LE link. Must be a multiple of 10 ms and larger than (1 + connPeripheralLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)
Min_CE_Length[i]	2	Information parameter about the minimum length for this LE connection. Time = N * 0.625 ms.	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)
Max_CE_Length[i]	2	Information parameter about the maximum length for this LE connection. Time = N * 0.625 ms.	0x0000 (0.000 ms) ... 0xFFFF (40959.375 ms)

Table 273. ACI_GAP_EXT_CREATE_CONNECTION output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_STATUS_EVENT
- HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT
- ACI_GAP_PROC_COMPLETE_EVENT

2.5 GATT/ATT commands

In Table 274 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 274. GATT/ATT commands list

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

Command	OpCode	LO	PO	BO	BF	LB
ACI_GATT_CONFIRM_INDICATION	0xFD25	-	-	-	Y	-
ACI_GATT_WRITE_RESP	0xFD26	-	Y	-	Y	-
ACI_GATT_ALLOW_READ	0xFD27	-	Y	-	Y	-
ACI_GATT_SET_SECURITY_PERMISSION	0xFD28	-	Y	-	Y	-
ACI_GATT_SET_DESC_VALUE	0xFD29	-	Y	-	Y	-
ACI_GATT_READ_HANDLE_VALUE	0xFD2A	-	Y	-	Y	-
ACI_GATT_UPDATE_CHAR_VALUE_EXT	0xFD2C	-	Y	-	Y	-
ACI_GATT_DENY_READ	0xFD2D	-	Y	-	Y	-
ACI_GATT_SET_ACCESS_PERMISSION	0xFD2E	-	Y	-	Y	-
ACI_GATT_STORE_DB	0xFD30	-	Y	-	Y	-
ACI_GATT_SET_MULT_NOTIFICATION	0xFD31	-	-	-	-	-
ACI_GATT_READ_MULTIPLE_VAR_CHAR_VALUE	0xFD32	-	-	-	-	-

2.5.1 ACI_GATT_INIT

Initializes the GATT layer for server and client roles, and adds the GATT service with service changed characteristic. Until this command is issued, the GATT channel does not process any command, even if the connection is opened. This command must be given before using any of the GAP features.

Input parameters:none

Table 275. 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

Adds 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 the Max_Attribute_Records parameter, which 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.

Table 276. ACI_GATT_ADD_SERVICE input parameters

Parameter	Size	Description	Possible values
Service_UUID_Type	1	UUID type (the selected value modifies the parameter Service_UUID)	• 0x01: 16-bit UUID • 0x02: 128-bit UUID
Service_UUID_16 or Service_UUID_128	2 or 16	16- 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	-

Table 277. **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

Includes 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 returns the handle of this attribute in Included_handle.

Table 278. **ACI_GATT_INCLUDE_SERVICE input parameters**

Parameter	Size	Description	Possible values
Service_Handle	2	Handle of the service to whom another service must be included	-
Include_Start_Handle	2	Start handle of the service that must be included in service	-
Include_End_Handle	2	End handle of the service that must 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- or 128-bit UUID	-

Table 279. **ACI_GATT_INCLUDE_SERVICE output parameters**

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

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.4 ACI_GATT_ADD_CHAR

This command adds a characteristic to a service, and returns the handle of the declaration attribute. The attribute that holds the characteristic value is always located at the next handle (Char_Handle + 1). The characteristic value is immediately followed, in order, by:

- the server characteristic configuration descriptor if CHAR_PROP_BROADCAST is selected
- the client characteristic configuration descriptor if CHAR_PROP_NOTIFY or CHAR_PROP_INDICATE is selected
- the characteristic extended properties descriptor if CHAR_PROP_EXT is selected

As an example, if CHAR_PROP_NOTIFY is selected, and CHAR_PROP_BROADCAST and CHAR_PROP_EXT are not, then the client characteristic configuration attribute handle is Char_Handle + 2.

Other descriptors can be added to the characteristic by calling the ACI_GATT_ADD_CHAR_DESC command immediately after this one.

Table 280. ACI_GATT_ADD_CHAR input parameters

Parameter	Size	Description	Possible values
Service_Handle	2	Handle of the service to whom the characteristic is added	-
Char_UUID_Type	1	UUID type	• 0x01 = 16-bit UUID • 0x02 = 128-bit UUID
Char_UUID	2 or 16	16- 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 without 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 • 0x08: GATT_NOTIFY_NOTIFICATION_COMPLETION
Enc_Key_Size	1	Minimum encryption key size required to read the characteristic	0x07 ... 0x10
Is_Variable	1	Specifies if the characteristic value has a fixed or a variable length	• 0x00: Fixed length • 0x01: Variable length

Table 281. **ACI_GATT_ADD_CHAR output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Char_Handle	2	Handle of the added characteristic. 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

Adds a characteristic descriptor to a service. This command allocates the new handle for the descriptor after the current one. It is advisable to call it following the call of ACI_GATT_ADD_CHAR that created the characteristic containing this descriptor.

Table 282. **ACI_GATT_ADD_CHAR_DESC input parameters**

Parameter	Size	Description	Possible values
Service_Handle	2	Handle of service to whom the characteristic belongs	-
Char_Handle	2	Handle of the characteristic to whom description must be added	-
Char_Desc_Uuid_Type	1	UUID type	• 0x01 = 16 bits • 0x02 = 128 bits
Char_Desc_Uuid	2 or 16	16- or 128-bit UUID	-
Char_Desc_Value _Max_Length	1	Maximum length of the descriptor value	-
Char_Desc _Value_Length	1	Current length of the characteristic description 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) • 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)

Parameter	Size	Description	Possible values
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 • 0x08: GATT_NOTIFY_NOTIFICATION_COMPLETION
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 or a variable length	• 0x00: Fixed length • 0x01: Variable length

Table 283. 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

Updates a characteristic value in a service. If notifications (or indications) are enabled on that characteristic, a notification (or indication) is sent to any client that has registered for notifications (or indications) via the Client Characteristic Configuration.

The command is not allowed if it causes the generation of an indication on a bearer still awaiting confirmation of a previous indication.

The command does not execute and returns BLE_STATUS_BUSY if notifications from a previous call are not completed. The application can enable and wait for ACI_GATT_NOTIFICATION_COMPLETE_EVENT to avoid this case.

The command does not execute and returns BLE_STATUS_INSUFFICIENT_RESOURCES if there is no more room in the TX pool to allocate notification (or indication) packets. This happens if notifications (or indications) are enabled and the application calls this command at a rate higher than what is allowed by the link. Throughput on BLE link depends on connection interval and connection length parameters (decided by the central, see ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_REQ for details on how to suggest new connection parameters from a peripheral). The application can wait for ACI_GATT_TX_POOL_AVAILABLE_EVENT before retrying a call to this command. It can also retry the call until it does not return BLE_STATUS_INSUFFICIENT_RESOURCES.

When calling this command, the characteristic value is updated only if the command returns BLE_STATUS_SUCCESS or BLE_STATUS_SEC_PERMISSION_ERROR. The security permission error means that at least one client has not been notified, because of security requirements not met.

Table 284. **ACI_GATT_UPDATE_CHAR_VALUE input parameters**

Parameter	Size	Description	Possible values
Service_Handle	2	Handle of service to whom the characteristic belongs	-
Char_Handle	2	Handle of the characteristic declaration	-
Val_Offset	1	The offset from which the attribute value must be updated. If this is set to 0 and the attribute value is of variable length, the length of the attribute is set to the Char_Value_Length. If the Val_Offset is set to a value greater than 0, 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. On STM32WB, cannot exceed (BLE_CMD_MAX_PARAM_LEN - 6), that is, 249 for the default value.	-
Char_Value	Char_Value_Length	Characteristic value	-

Table 285. **ACI_GATT_UPDATE_CHAR_VALUE output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- ACI_GATT_NOTIFICATION_COMPLETE_EVENT

2.5.7 ACI_GATT_DEL_CHAR

Deletes the specified characteristic from the service.

Table 286. **ACI_GATT_DEL_CHAR input parameters**

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

Output parameters

Table 287. **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

Deletes the specified service from the GATT server database.

Table 288. **ACI_GATT_DEL_SERVICE input parameters**

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

Table 289. **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

Deletes the include definition from the service.

Table 290. **ACI_GATT_DEL_INCLUDE_SERVICE input parameters**

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

Table 291. **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

Masks events from the GATT. If the bit in GATT_Evt_Mask is set to 1, the event associated with that bit is enabled. In the default configuration all the events are masked.

Table 292. **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

Table 293. **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

Performs an ATT MTU exchange procedure. When the ATT MTU exchange procedure is completed, an ACI_ATT_EXCHANGE_MTU_RESP_EVENT is generated. An ACI_GATT_PROC_COMPLETE_EVENT is also generated to indicate the end of the procedure.

Table 294. **ACI_GATT_EXCHANGE_CONFIG input parameters**

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

Table 295. **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

Sends 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. The end of the procedure is indicated by an ACI_GATT_PROC_COMPLETE_EVENT.

Table 296. **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	-

Table 297. **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

Sends a find by type value request, 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. The end of the procedure is indicated by an ACI_GATT_PROC_COMPLETE_EVENT.

Table 298. 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	-

Table 299. 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

Sends 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. The responses are given through the ACI_ATT_READ_BY_TYPE_RESP_EVENT.

Table 300. 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- or 128-bit UUID	-

Table 301. 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

Sends a read by group type request, used to obtain the values of grouping attributes where the attribute type is known but the handle is not. 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. The end of the procedure is indicated by an ACI_GATT_PROC_COMPLETE_EVENT.

Table 302. 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- or 128-bit UUID	-

Table 303. 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

Sends a prepare write request, 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. The end of the procedure is indicated by an ACI_GATT_PROC_COMPLETE_EVENT.

Table 304. 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	-

Table 305. 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

Sends 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. The end of the procedure is indicated by an ACI_GATT_PROC_COMPLETE_EVENT.

Table 306. 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

Table 307. 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

Starts 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.

Table 308. ACI_GATT_DISC_ALL_PRIMARY_SERVICES input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)

Table 309. 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

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

Table 310. ACI_GATT_DISC_PRIMARY_SERVICE_BY_UUID input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
UUID_Type	1	UUID type: 0x01 = 16 bits UUID while 0x02 = 128 bits UUID	-
UUID	2 or 16	16- or 128-bit UUID	-

Output parameters

Table 311. 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

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

Table 312. ACI_GATT_FIND_INCLUDED_SERVICES input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Start_Handle	2	Start attribute handle of the service	-
End_Handle	2	End attribute handle of the service	-

Output parameters

Table 313. 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

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

Table 314. 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: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Start_Handle	2	Start attribute handle of the service	-
End_Handle	2	End attribute handle of the service	-

Table 315. 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

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

Table 316. ACI_GATT_DISC_CHAR_BY_UUID input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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- or 128-bit UUID	-

Output parameters

Table 317. 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

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

Table 318. ACI_GATT_DISC_ALL_CHAR_DESC input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Char_Handle	2	Handle of the characteristic value	-
End_Handle	2	End handle of the characteristic	-

Table 319. 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

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

Table 320. **ACI_GATT_READ_CHAR_VALUE input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the characteristic value to be read	-

Table 321. **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

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 is generated. Before completion, the response packets are given through an ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT.

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

Table 322. **ACI_GATT_READ_USING_CHAR_UUID input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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- or 128-bit UUID	-

Output parameters

Table 323. **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

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

Table 324. ACI_GATT_READ_LONG_CHAR_VALUE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the characteristic value to be read	-
Val_Offset	2	Offset from which the value must be read	-

Table 325. 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

Starts 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, an ACI_GATT_PROC_COMPLETE_EVENT is generated. Before completion, the response packets are given through an ACI_ATT_READ_MULTIPLE_RESP_EVENT.

Table 326. ACI_GATT_READ_MULTIPLE_CHAR_VALUE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Number_of_Handles	1	The number of handles for which the value must be read	-
Handle[i]	2	The handles for which the attribute value must be read	-

Table 327. **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

Starts the procedure to write a characteristic value. When the procedure is completed, an ACI_GATT_PROC_COMPLETE_EVENT is generated.

Table 328. **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: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the characteristic value to be written	-
Attribute_Val_Length	1	Length of the value to be written. On STM32WB, it cannot exceed (BLE_CMD_MAX_PARAM_LEN - 5), that is, 250 for the default value.	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Table 329. **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

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

Table 330. ACI_GATT_WRITE_LONG_CHAR_VALUE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the characteristic value to be written	-
Val_Offset	2	Offset at which the attribute must be written	-
Attribute_Val_Length	1	Length of the value to be written. On STM32WB, cannot exceed (BLE_CMD_MAX_PARAM_LEN - 7), that is, 248 for the default value.	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Table 331. 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

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

Table 332. **ACI_GATT_WRITE_CHAR_RELIABLE input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the attribute to be written	-
Val_Offset	2	Offset at which the attribute must be written	-
Attribute_Val_Length	1	Length of the value to be written. On STM32WB, cannot exceed (BLE_CMD_MAX_PARAM_LEN - 7), that is, 248 for the default value.	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Table 333. **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

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

Table 334. **ACI_GATT_WRITE_LONG_CHAR_DESC input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the attribute to be written	-
Val_Offset	2	Offset at which the attribute musto be written	-
Attribute_Val_Length	1	Length of the value to be written. On STM32WB, cannot exceed (BLE_CMD_MAX_PARAM_LEN - 7), that is, 248 for the default value.	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Table 335. **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

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

Table 336. **ACI_GATT_READ_LONG_CHAR_DESC input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the characteristic descriptor	-
Val_Offset	2	Offset from which the value must be read	-

Table 337. **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

Starts the procedure to write a characteristic descriptor. When completed, an ACI_GATT_PROC_COMPLETE_EVENT is generated.

Table 338. ACI_GATT_WRITE_CHAR_DESC input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the attribute to be written	-
Attribute_Val_Length	1	Length of the value to be written, cannot exceed (ATT_MTU - 3). On STM32WB, it cannot exceed (BLE_CMD_MAX_PARAM_LEN - 5), that is, 250 for the default value.	-
Attribute_Val	Attribute_Val_Length	Value to be written	-

Table 339. 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

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

Table 340. ACI_GATT_READ_CHAR_DESC input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the descriptor to be read	-

Table 341. **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

Starts the procedure to write a characteristic value without waiting for responses from the server. No events are generated after the command is executed.

The length of the value to be written must not exceed neither ATT_MTU - 3, nor BLE_EVT_MAX_PARAM_LEN - 5 (250 for the BLE_EVT_MAX_PARAM_LEN default value).

Table 342. **ACI_GATT_WRITE_WITHOUT_RESP input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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	-

Table 343. **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

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

The length of the value to be written must not exceed neither ATT_MTU - 15, nor BLE_EVT_MAX_PARAM_LEN - 5 (250 for BLE_EVT_MAX_PARAM_LEN default value).

Table 344. **ACI_GATT_SIGNED_WRITE_WITHOUT_RESP input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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	-

Table 345. **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

This command must be sent when the application receives the ACI_GATT_INDICATION_EVENT, to confirm the indication.

Table 346. **ACI_GATT_CONFIRM_INDICATION input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)

Table 347. **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

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

Table 348. **ACI_GATT_WRITE_RESP input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the attribute passed in the event EVT_BLUE_GATT_WRITE_PERMIT_REQ	-
Write_status	1	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 passed to the client if the write must 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	-

Table 349. **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

Allows the GATT server to send a response to a read request from a client. The application must send this command when it receives the ACI_GATT_READ_PERMIT_REQ_EVENT or ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT. The command indicates to the stack that the response can be sent to the client. If the application must update any of the attributes before they are read by the client, it must update the characteristic values using the ACI_GATT_UPDATE_CHAR_VALUE, and then issue the command. The application must perform the required operations within 30 seconds, otherwise the GATT procedure is timed out.

Table 350. **ACI_GATT_ALLOW_READ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)

Table 351. **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

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

Table 352. **ACI_GATT_SET_SECURITY_PERMISSION input parameters**

Parameter	Size	Description	Possible values
Serv_Handle	2	Handle of the service containing 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 (needs authentication to read) • 0x02: AUTHOR_READ (needs authorization to read) • 0x04: ENCRY_READ (needs encryption to read) • 0x08: AUTHEN_WRITE (needs authentication to write) • 0x10: AUTHOR_WRITE (needs authorization to write) • 0x20: ENCRY_WRITE (needs encryption to write)

Table 353. **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

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

Table 354. **ACI_GATT_SET_DESC_VALUE input parameters**

Parameter	Size	Description	Possible values
Serv_Handle	2	Handle of the service containing the characteristic descriptor	-
Char_Handle	2	Handle of the characteristic containing the descriptor	-
Char_Desc_Handle	2	Handle of the descriptor whose value has to be set	-

Parameter	Size	Description	Possible values
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	-

Table 355. 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.

Table 356. 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	-

Table 357. 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

This is a more flexible version of the ACI_GATT_UPDATE_CHAR_VALUE command. It supports the update of long attributes (up to 512 bytes), and indicates selectively the generation of indication/notification.

Table 358. **ACI_GATT_UPDATE_CHAR_VALUE_EXT input parameters**

Parameter	Size	Description	Possible values
Conn_Handle_To_Notify	2	Specifies the client(s) to be notified.	• 0x0000: Notify all subscribed clients on their unenhanced ATT bearer • 0x0001 ... 0x0EFF: Notify one client on the specified unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: Notify one client on the specified enhanced ATT bearer (the LSB of the parameter is the connection oriented channel index)
Service_Handle	2	Handle of service to whom the characteristic belongs	-
Char_Handle	2	Handle of the characteristic declaration	-
Update_Type	1	Allows 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. For a variable size characteristic, this field specifies the new length of the characteristic value after the update; in case of fixed length, this field is ignored.	-
Value_Offset	2	The offset from which the attribute value must be updated	-
Value_Length	1	Length of the value parameter in octets. On STM32WB, cannot exceed (BLE_CMD_MAX_PARAM_LEN - 12), that is, 243 for the default value.	-
Value	Value_Length	Updated characteristic value	-

Table 359. **ACI_GATT_UPDATE_CHAR_VALUE_EXT output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- ACI_GATT_NOTIFICATION_COMPLETE_EVENT

2.5.44 ACI_GATT_DENY_READ

Forbids the GATT server to send a response to a read request from a client. The application can send this command when it receives the ACI_GATT_READ_PERMIT_REQ_EVENT or ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT. The command indicates to the stack that the client is not allowed to read the requested characteristic (for example, due to 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 ACI_GATT_READ_PERMIT_REQ_EVENT or ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT, otherwise the GATT procedure issues a timeout.

Table 360. **ACI_GATT_DENY_READ input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Error_Code	1	Error code for the command	• 0x08: Insufficient authorization • 0x80 ... 0x9F: application error

Table 361. **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

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

Table 362. **ACI_GATT_SET_ACCESS_PERMISSION input parameters**

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

Table 363. **ACI_GATT_SET_ACCESS_PERMISSION output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.46 ACI_GATT_STORE_DB

This command forces the saving of the GATT database for all active connections (by default the database is saved per active connection at the time of disconnection).

Input parameters: none

Table 364. **ACI_GATT_STORE_DB output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.47 ACI_GATT_SET_MULT_NOTIFICATION

This command sends a Multiple Handle Value Notification over the specified ATT bearer. The handles provided as parameters must be those of the characteristics declaration.

Table 365. ACI_GATT_SET_MULT_NOTIFICATION input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Number_Of_Handles	1	Number of handles in the next table	0x02 ... 0x7E
Handle[I]	2	Attribute handle	-

Table 366. ACI_GATT_SET_MULT_NOTIFICATION output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.5.48 ACI_GATT_READ_MULTIPLE_VAR_CHAR_VALUE

Starts a procedure for reading multiple variable length characteristic values from a server. The command must specify the handles of the characteristic values to be read. When the procedure is completed, an ACI_GATT_PROC_COMPLETE_EVENT is generated. Before completion, the response packets are given through the ACI_ATT_READ_MULTIPLE_RESP_EVENT.

Table 367. ACI_GATT_READ_MULTIPLE_VAR_CHAR_VALUE input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Number_Of_Handles	1	Number of handles in the next table	0x02 ... 0x7E
Handle[I]	2	Attribute handle	-

Table 368. ACI_GATT_READ_MULTIPLE_VAR_CHAR_VALUE output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.6 L2CAP commands

In Table 369 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 369. L2CAP commands list

Command	OpCode	LO	PO	BO	BF	LB
ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_REQ	0xFD81	-	Y	-	Y	-
ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_RESP	0xFD82	-	-	-	Y	-
ACI_L2CAP_COC_CONNECT	0xFD88	-	-	-	-	-
ACI_L2CAP_COC_CONNECT_CONFIRM	0xFD89	-	-	-	-	-
ACI_L2CAP_COC_RECONF	0xFD8A	-	-	-	-	-
ACI_L2CAP_COC_RECONF_CONFIRM	0xFD8B	-	-	-	-	-
ACI_L2CAP_COC_DISCONNECT	0xFD8C	-	-	-	-	-
ACI_L2CAP_COC_FLOW_CONTROL	0xFD8D	-	-	-	-	-
ACI_L2CAP_COC_TX_DATA	0xFD8E	-	-	-	-	-

2.6.1 ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_REQ

Sends an L2CAP connection parameter update request from the peripheral to the central. An ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT is raised when the central responds (accepts or rejects) to the request.

Table 370. 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)
Conn_latency	2	Maximum peripheral latency for the connection, in number of connection events.	0x0000 ... 0x01F3
Timeout_Multiplier	2	Defines connection timeout parameter as: Timeout Multiplier * 10ms.	-

Table 371. 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

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

Table 372. 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)
Conn_latency	2	Maximum peripheral latency for the connection, in number of connection events.	0x0000 ... 0x01F3
Timeout_Multiplier	2	Defines connection timeout parameter as: Timeout Multiplier * 10 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)
Identifier	1	Identifier received in ACI_L2CAP_CONNECTION_UPDATE_REQ EVENT	-
Accept	1	Specifies if connection update parameters are acceptable or not.	• 0x00: Reject • 0x01: Accept

Table 373. ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_RESP output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.6.3 ACI_L2CAP_COC_CONNECT

This command sends a credit based connection request packet on the specified connection. See Bluetooth core specification Vol.3 Part A.

Table 374. ACI_L2CAP_COC_CONNECT input parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle to whom the command applies	0x0000 ... 0x0EFF
SPSM	2	Simplified Protocol/Service Multiplexer	0x0001 ... 0x00FF
MTU	2	Maximum transmission unit	23 ... 65535
MPS	2	Maximum payload size, in octects	23 ... 248
Initial_Credits	2	Number of K-frames that can be received on the created channel(s) by the L2CAP layer entity sending this packet	23 ... 65535
Channel_Number	1	Number of channels to be created. If this parameter is set to 0, it requests the creation of one LE credit based connection-oriented channel, otherwise it requests the creation of one or more enhanced credit based connection-oriented channels.	0 ... 5

Table 375. **ACI_L2CAP_COC_CONNECT output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- ACI_L2CAP_COC_CONNECT_CONFIRM_EVENT
- ACI_L2CAP_PROC_TIMEOUT_EVENT
- ACI_L2CAP_COMMAND_REJECT_EVENT

2.6.4 ACI_L2CAP_COC_CONNECT_CONFIRM

This command sends a credit based connection response packet. It must be used upon receipt of a connection request through an ACI_L2CAP_COC_CONNECT_EVENT. See Bluetooth core specification Vol.3 Part A.

Table 376. **ACI_L2CAP_COC_CONNECT_CONFIRM input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the command applies	0x0000 ... 0x0EFF
MTU	2	Maximum transmission unit	23 ... 65535
MPS	2	Maximum payload size, in octects	23 ... 248
Initial_Credits	2	Number of K-frames that can be received on the created channel(s) by the L2CAP layer entity sending this packet	23 ... 65535
Results	2	This parameter indicates the outcome of the request. A value of 0x0000 indicates success, a non-zero value indicates that the request is refused.	0x0000 ... 0x000C

Table 377. **ACI_L2CAP_COC_CONNECT_CONFIRM output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-
Channel_Number	1	Number of created channels (it is the length of Channel_Index_List)	0 ... 5
Channel_Index_List	Channel_Number	List of channel indexes to whom the primitive applies	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.6.5 ACI_L2CAP_COC_RECONF

This command sends a credit based reconfigure request packet on the specified connection. See Bluetooth core specification Vol.3 Part A.

Table 378. **ACI_L2CAP_COC_RECONF input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle to whom the command applies	0x0000 ... 0x0EFF
MTU	2	Maximum transmission unit	23 ... 65535
MPS	2	Maximum payload size in octects	23 ... 248
Channel_Number	2	Number of channels to be created (it is the length of Channel_Index_List)	1 ... 5
Channel_Index_List	Channel_Number	List of channel indexes to whom the primitive applies.	-

Table 379. **ACI_L2CAP_COC_RECONF output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- ACI_L2CAP_COC_CONNECT_CONFIRM_EVENT
- ACI_L2CAP_PROC_TIMEOUT_EVENT
- ACI_L2CAP_COMMAND_REJECT_EVENT

2.6.6 ACI_L2CAP_COC_RECONF_CONFIRM

This command sends a credit based reconfigure response packet. It must be used upon receipt of a credit based reconfigure request through an ACI_L2CAP_COC_RECONF_EVENT. See Bluetooth core specification Vol.3 Part A.

Table 380. **ACI_L2CAP_COC_RECONF_CONFIRM input parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle to whom the command applies	0x0000 ... 0x0EFF
Results	2	This parameter indicates the outcome of the request. 0x0000 indicates success, while a non-zero value indicates that the request is refused.	0x0000 ... 0x000C

Table 381. **ACI_L2CAP_COC_RECONF_CONFIRM output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.6.7 ACI_L2CAP_COC_DISCONNECT

This command sends a disconnection request signaling packet on the specified connectionoriented channel. See Bluetooth core specification Vol.3 Part A.

Table 382. **ACI_L2CAP_COC_DISCONNECT input parameters**

Parameter	Size	Description	Possible values
Channel_Index	1	Index of the connection-oriented channel to whom the command applies	-

Output parameters

Table 383. **ACI_L2CAP_COC_DISCONNECT output parameters**

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT
- ACI_L2CAP_COC_CONNECT_CONFIRM_EVENT
- ACI_L2CAP_PROC_TIMEOUT_EVENT

- ACI_L2CAP_COMMAND_REJECT_EVENT

2.6.8 ACI_L2CAP_COC_FLOW_CONTROL

This command sends a flow control credit signaling packet on the specified connection-oriented channel. See Bluetooth core specification Vol.3 Part A.

Table 384. ACI_L2CAP_COC_FLOW_CONTROL input parameters

Parameter	Size	Description	Possible values
Channel_Index	1	Index of the connection-oriented channel to whom the primitive applies.	-
Credits	2	Number of credits the receiving device can increment, corresponding to the number of K-frames that can be sent to the peer device sending the Flow Control Credit packet.	1 ... 65535

Table 385. ACI_L2CAP_COC_FLOW_CONTROL output parameters

Parameter	Size	Description	Possible values
Status	1	Status error code	-

Events generated

- HCI_COMMAND_COMPLETE_EVENT

2.6.9 ACI_L2CAP_COC_TX_DATA

This command sends a K-frame packet on the specified connection-oriented channel. See Bluetooth core specification Vol.3 Part A.

Note: For the first K-frame of the SDU, the information data must contain the L2CAP SDU length coded on two octets, followed by the K-frame information payload. For the next K-frames, the information data must contain only the K-frame information payload. The Length value must not exceed (BLE_CMD_MAX_PARAM_LEN - 3), i.e. 252 for BLE_CMD_MAX_PARAM_LEN default value.

Table 386. ACI_L2CAP_COC_TX_DATA input parameters

Parameter	Size	Description	Possible values
Channel_Index	1	Index of the connection-oriented channel to whom the primitive applies	-
Length	2	Length of data, in octects	-
Data	Length	Information data	-

Table 387. ACI_L2CAP_COC_TX_DATA 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 388 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 388. HCI events commands list

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

3.1.1 HCI_DISCONNECTION_COMPLETE_EVENT

This 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, it can be ignored by the host.

If the connection is terminated by the remote device, the Reason parameter is set to the reason specified by the remote device only if it has an allowed value, otherwise it is forced to Remote user terminated connection error code (0x13). Allowed remote Reason values are:

- Authentication failure error code (0x05)
- Other end terminated connection error codes (0x13 to 0x15)
- Unsupported remote feature error code (0x1A)
- Unacceptable connection parameters error code (0x3B)

Table 389. HCI_DISCONNECTION_COMPLETE_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

This event is used to indicate that the change of the encryption mode has been completed.

The Connection_Handle is for an ACL connection. The Encryption_Enabled event parameter specifies the new Encryption_Enabled parameter for the Connection_Handle specified by the event parameter. This event occurs on both devices to notify the hosts when encryption has changed for the specified Connection_Handle between two devices. 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 generated if encryption is paused or resumed (for example, during a role switch).

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 uses only Encryption_Enabled values 0x00 (OFF) and 0x01 (ON).

Table 390. HCI_ENCRYPTION_CHANGE_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

This 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 changes 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.

Table 391. HCI_READ_REMOTE_VERSION_INFORMATION_COMPLETE_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

This event is used to indicate the implementation specific type of hardware failure for the controller. It notifies the host that a hardware failure has occurred in the controller.

Table 392. HCI_HARDWARE_ERROR_EVENT parameters

Parameter	Size	Description	Possible values
Hardware_Code	1	Hardware error event code: • 0: not used • 1: bluecore act2 error • 2: bluecore time overrun error • 3: internal FIFO full • 4: ISR delay error	• 0x00: not used • 0x01: event_act2 error • 0x02: event_time_overrun error • 0x03: event_fifo_full error • 0x04: event_isr_delay error

3.1.5 HCI_NUMBER_OF_COMPLETED_PACKETS_EVENT

This 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 on 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.

Table 393. HCI_NUMBER_OF_COMPLETED_PACKETS_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

This event is used to indicate to the Host that the encryption key was refreshed on the given Connection_Handle. The controller sends this event when the encryption key has been refreshed, because the encryption started or resumed.

Table 394. HCI_ENCRYPTION_KEY_REFRESH_COMPLETE_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

This 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 that 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 0. 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.

Table 395. HCI_COMMAND_COMPLETE_EVENT parameters

Parameter	Size	Description	Possible values
Num_HCI_Command_Packets	1	The number of HCI command packets allowed to be sent to the controller from the host.	-
Command_Opcode	2	Opcode of the command which caused this event	-
Return_Parameters	Variable	Return parameter(s) for the command specified in the Command_Opcode event parameter.	-

3.1.8 HCI_COMMAND_STATUS_EVENT

This 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.

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, this parameter is set to 0. 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.

Table 396. HCI_COMMAND_STATUS_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 397 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / PO / BF/ LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 397. HCI LE META events command list

Command	OpCode	LO	PO	BO	BF	LB
HCI_LE_CONNECTION_COMPLETE_EVENT	0x01	Y	Y	-	Y	Y
HCI_LE_ADVERTISING_REPORT_EVENT	0x02	Y	-	Y	Y	Y
HCI_LE_CONNECTION_UPDATE_COMPLETE_EVENT	0x03	Y	Y	-	Y	Y
HCI_LE_READ_REMOTE_FEATURES_COMPLETE_EVENT	0x04	Y	Y	-	Y	Y
HCI_LE_LONG_TERM_KEY_REQUEST_EVENT	0x05	Y	Y	-	Y	Y
HCI_LE_DATA_LENGTH_CHANGE_EVENT	0x07	Y	Y	-	Y	Y
HCI_LE_READ_LOCAL_P256_PUBLIC_KEY_COMPLETE_EVENT	0x08	Y	Y	-	Y	Y
HCI_LE_GENERATE_DHKEY_COMPLETE_EVENT	0x09	Y	-	-	-	Y
HCI_LE_ENHANCED_CONNECTION_COMPLETE_EVENT	0x0A	Y	Y	-	Y	Y
HCI_LE_DIRECTED_ADVERTISING_REPORT_EVENT	0x0B	Y	-	-	Y	Y
HCI_LE_PHY_UPDATE_COMPLETE_EVENT	0x0C	Y	-	-	Y	Y
HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT	0x0D	Y	-	-	-	-
HCI_LE_SCAN_TIMEOUT_EVENT	0x11	Y	-	-	-	-
HCI_LE_ADVERTISING_SET_TERMINATED_EVENT	0x12	Y	-	-	-	-
HCI_LE_SCAN_REQUEST_RECEIVED_EVENT	0x13	Y	-	-	-	-
HCI_LE_CHANNEL_SELECTION_ALGORITHM_EVENT	0x14	Y	Y	-	-	-

3.2.1 HCI_LE_CONNECTION_COMPLETE_EVENT

This event indicates to both the hosts forming the connection that a new connection has been created. Upon the creation, 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 issued the HCI_LE_CREATE_CONNECTION command to indicates if the connection establishment failed or was successful. The Central_Clock_Accuracy parameter is only valid for a peripheral. On a central, this parameter is set to 0x00.

Table 398. HCI_LE_CONNECTION_COMPLETE_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: Central • 0x01: Peripheral
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)

Parameter	Size	Description	Possible values
Conn_Latency	2	Maximum peripheral 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 + connPeripheralLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000)
Central_Clock_Accuracy	1	Central clock accuracy, only valid for a peripheral	• 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

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).

Table 399. HCI_LE_ADVERTISING_REPORT_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	Address type	• 0x00: Public device address • 0x01: Random device address • 0x02: Public identity address • 0x03: Random (static) identity address
Address[i]	6	Public or random device address of the device to be connected	-
Length_Data[i]	1	Length of the Data[i] field for each device who 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), in dBm	• 127: RSSI not available • -127 ... 20

3.2.3 HCI_LE_CONNECTION_UPDATE_COMPLETE_EVENT

This event is used to indicate that the controller process to update the connection has completed. On a peripheral, if no connection parameters are updated, this event is not issued. On a central, this event is issued if the Connection_Update command was sent.

Table 400. **HCI_LE_CONNECTION_UPDATE_COMPLETE_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	Maximum peripheral 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 + connPeripheralLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)

3.2.4 HCI_LE_READ_REMOTE_FEATURES_COMPLETE_EVENT

Description

This 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.

Table 401. **HCI_LE_READ_REMOTE_FEATURES_COMPLETE_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

Indicates that the central device is attempting to encrypt or re-encrypt the link, and is requesting the long term key from the host.

Table 402. **HCI_LE_LONG_TERM_KEY_REQUEST_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

Notifies the host of a change in either the maximum payload length, or the maximum transmission time of packets, in both directions.

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

Table 403. **HCI_LE_DATA_LENGTH_CHANGE_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	Maximum number of payload octets in a link layer packet that the local controller sends on this connection (connEffectiveMaxTxOctets)	0x001B ... 0x00FB
MaxTxTime	2	Maximum time that the local controller takes to send a link layer packet on this connection (connEffectiveMaxTxTime)	0x0148 ... 0x4290
MaxRxOctets	2	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	Maximum time that the local controller expects to receive a link layer packet on this connection (connEffectiveMaxRxTime)	0x0148 ... 0x4290

3.2.7 HCI_LE_READ_LOCAL_P256_PUBLIC_KEY_COMPLETE_EVENT

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

Table 404. **HCI_LE_READ_LOCAL_P256_PUBLIC_KEY_COMPLETE_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

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

Table 405. **HCI_LE_GENERATE_DHKEY_COMPLETE_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

Indicates to both the hosts that a new connection has been created. Upon its creation, a Connection_Handle is assigned by the controller, and passed to the host. If the connection set-up fails, this event is provided to the host that issued the LE_Create_Connection command. If this event and the LE connection complete events are 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 set-up failed or was successful. The Central_Clock_Accuracy parameter is only valid for a peripheral. On a central, this parameter is set to 0x00.

Table 406. **HCI_LE_ENHANCED_CONNECTION_COMPLETE_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: Central • 0x01: Peripheral
Peer_Address_Type	1	0x00 Public device address 0x01 Random device address	• 0x00: Public device address

Parameter	Size	Description	Possible values
		0x02 Public identity address (corresponds to resolved private address) 0x03 Random (static) identity address (corresponds to resolved private address)	• 0x01: Random device address • 0x02: Public identity address • 0x03: Random (static) identity address
Peer_Address	6	Public or random device address, public 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 0s.	-
Peer_Resolvable_Private_Address	6	Resolvable private address used by the peer device for this connection. This is only valid for Peer_Address_Type 0x02 and 0x03. For other values the controller returns all 0s.	-
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	Maximum peripheral 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 + connPeripheralLatency) * connInterval * 2. Time = N * 10 ms	0x000A (100 ms) ... 0x0C80 (32000 ms)
Central_Clock_Accuracy	1	Central clock accuracy, valid for a peripheral	• 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_DIRECTED_ADVERTISING_REPORT_EVENT

This vent 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 identify the address the directed advertisements are being directed to. Address_Type and Address are the address of the advertiser sending the directed advertisements.

Table 407. **HCI_LE_DIRECTED_ADVERTISING_REPORT_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	Address type	• 0x00: public device address • 0x01: random device address • 0x02: public identity address • 0x03: random (static) identity address
Address[i]	6	Public or random device address, public or random (static) identity address of the advertising device	-
Direct_Address_Type[i]	1	Address type	0x01: random device address
Direct_Address[i]	6	Random device address	-
RSSI[i]	1	N Size: 1 octet (signed integer), in dBm	• 127: RSSI not available • -127 ... 20

3.2.11 HCI_LE_PHY_UPDATE_COMPLETE_EVENT

This event is issued to indicate that the controller has changed the transmitter PHY in use, or the receiver PHY in use, or both. If an LE_Set_PHY command was sent and the controller determines that no PHYs change as a result, it issues this event immediately.

Table 408. **HCI_LE_PHY_UPDATE_COMPLETE_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)

3.2.12 HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT

This event indicates that a Bluetooth device has responded to an active scan or has broadcast advertisements received during a passive scan. See Bluetooth spec. v.5.2 [Vol 4, Part E, 7.7.65.13].

Table 409. **HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT parameters**

Parameter	Size	Description	Possible values
Num_Reports	1	Number of responses	0x01
Event_Type	1	Event type	Bitmask of: • 0x0001: connectable advertising • 0x0002: scannable advertising • 0x0004: directed advertising • 0x0008: scan response • 0x0010: legacy advertising

Parameter	Size	Description	Possible values
			PDUs used: • 0x0020: incomplete, more data to come • 0x0040: incomplete, data truncated, no more to come
Address_Type	1	Address type of the advertising device	• 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) • 0xFF: no address provided (anonymous advertisement)
Address	6	Public, random, public or random (static) identity address of the advertising device	-
Primary_PHY	1	Primary advetising PHY	• 0x01: advertiser PHY is LE 1M
Secondary_PHY	1	Secondary advetising PHY	• 0x00: no packets on the secondary advertising physical channel • 0x01: advertiser PHY is LE 1M • 0x02: advertiser PHY is LE 2M • 0x01: advertiser PHY is LE coded
Advertising_SID	1	Value of the Advertising SID subfield in the ADI field of the PDU or, for scan responses, in the ADI field of the original scannable.	• 0xFF: no ADI field provided • 0x00 ... 0x0F: advertising SID subfield
TX_Power	1	Tx Power (signed integer), in dBm	• 127: Tx power information not available • -127 ... 20: Tx power
RSSI	1	RSSI (signed integer), in dBm	• 127: RSSI not available • -127 ... 20: Tx power
Periodic_Adv_Interval	2	Interval of the periodic advertising	• 0x0000: no periodic advertising
Direct_Address_Type	1	Target device address type	• 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) • 0xFE: random device address (controller unable to resolve)
Direct_Address	6	Public, random, public or random (static) identity address of the advertising device	-
Data_Length	1	Length of data	-
Data	Data_Length	Octets of advertising or scan response data formatted as defined in Bluetooth spec. v.5.2 [Vol 3, Part C, 11]	-

3.2.13 HCI_LE_SCAN_TIMEOUT_EVENT

This event indicates that scanning has ended because the duration has expired. See Bluetooth spec. v.5.2 [Vol 4, Part E, 7.7.65.17].

Event parameters: none

3.2.14 HCI_LE_ADVERTISING_SET_TERMINATED_EVENT

This event indicates that the controller has terminated advertising in the sets specified by the Advertising_Handle parameter. See Bluetooth spec. v.5.2 [Vol 4, Part E, 7.7.65.18].

Table 410. HCI_LE_ADVERTISING_SET_TERMINATED_EVENT parameters

Parameter	Size	Description	Possible values
Status	1	Error code	-
Advertising_Handle	1	Used to identify an advertising set	0x00 ... 0xEF
Connection_Handle	2	Connection handle for which the event applies	0x0000 ... 0x0EFF
Num_Completed_Ext_Adv_Events	1	Number of completed extended advertising events transmitted by the controller	0x00 ... 0xFF

3.2.15 HCI_LE_SCAN_REQUEST_RECEIVED_EVENT

Description

This event indicates that a SCAN_REQ PDU or an AUX_SCAN_REQ PDU has been received by the advertiser. The request contains a device address from a scanner allowed by the advertising filter policy. The advertising set is identified by Advertising_Handle. See Bluetooth spec. v.5.2 [Vol 4, Part E, 7.7.65.19].

Table 411. HCI_LE_SCAN_REQUEST_RECEIVED_EVENT parameters

Parameter	Size	Description	Possible values
Advertising_Handle	1	Used to identify an advertising set	0x00 ... 0xEF
Scanner_Address_Type	1	Scanner address type	• 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) PDUs used: • 0x0020: incomplete, more data to come • 0x0040: incomplete, data truncated, no more to come
Scanner_Address	6	Public, random, public or random (static) identity address of the advertising device	-

3.2.16 HCI_LE_CHANNEL_SELECTION_ALGORITHM_EVENT

This event indicates which channel selection algorithm is used on a data physical channel connection. See Bluetooth spec. v.5.2 [Vol 4, Part E, 7.7.65.20].

Table 412. HCI_LE_CHANNEL_SELECTION_ALGORITHM_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
Channel_Selection_Algorithm	1	LE channel selection algorithm	• 0x00: algorithm #1 is used • 0x01: algorithm #2 is used

3.3 ACI GAP events

In Table 413 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO /BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only- basic.

Table 413. ACI GAP events commands list

Command	OpCode	LO	PO	BO	BF	LB
ACI_GAP_LIMITED_DISCOVERABLE_EVENT	0x0400	-	Y	-	Y	-
ACI_GAP_PAIRING_COMPLETE_EVENT	0x0401	-	Y	-	Y	-
ACI_GAP_PASS_KEY_REQ_EVENT	0x0402	-	Y	-	Y	-
ACI_GAP_AUTHORIZATION_REQ_EVENT	0x0403	-	Y	-	Y	-
ACI_GAP_PERIPHERAL_SECURITY_INITIATED_EVENT	0x0404	-	Y	-	Y	-
ACI_GAP_BOND_LOST_EVENT	0x0405	-	Y	-	Y	-
ACI_GAP_PROC_COMPLETE_EVENT	0x0407	-	Y	-	Y	-
ACI_GAP_ADDR_NOT_RESOLVED_EVENT	0x0408	-	Y	-	Y	-
ACI_GAP_NUMERIC_COMPARISON_VALUE_EVENT	0x0409	-	Y	-	Y	-
ACI_GAP_KEYPRESS_NOTIFICATION_EVENT	0x040A	-	Y	-	Y	-
ACI_GAP_PAIRING_REQUEST_EVENT	0x040B	-	Y	-	Y	-

3.3.1 ACI_GAP_LIMITED_DISCOVERABLE_EVENT

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

Event parameters: none

3.3.2 ACI_GAP_PAIRING_COMPLETE_EVENT

This event is generated when the pairing process completes successfully, or a pairing procedure timeout occurs, or the pairing fails. The event notifies 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).

Table 414. ACI_GAP_PAIRING_COMPLETE_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 • 0x03: Encryption failed
Reason	1	Pairing reason error code (valid in case of pairing failed status, else refer to Section 4: Status error codes)	• 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

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.

Table 415. ACI_GAP_PASS_KEY_REQ_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

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

Table 416. ACI_GAP_AUTHORIZATION_REQ_EVENT parameters

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

3.3.5 ACI_GAP_PERIPHERAL_SECURITY_INITIATED_EVENT

This event is generated when the peripheral security request is successfully sent to the central.

Event parameters: none

3.3.6 ACI_GAP_BOND_LOST_EVENT

This event is generated when a pairing request is issued in response to a peripheral security request from a central previously bonded with the peripheral. When this event is received, the upper layer must issue the command ACI_GAP_ALLOW_REBOND to allow the peripheral to continue the pairing process with the central.

Event parameters: none

3.3.7 ACI_GAP_PROC_COMPLETE_EVENT

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.

Table 417. ACI_GAP_PROC_COMPLETE_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	-

Parameter	Size	Description	Possible values
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

This event is sent only by a privacy enabled peripheral with a non empty bonded device list, when the peripheral is unsuccessful in resolving the resolvable address of the peer device after connecting to it.

Table 418. ACI_GAP_ADDR_NOT_RESOLVED_EVENT parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle for which the private address has not been resolved with any of the stored IRKs	-

3.3.9 ACI_GAP_NUMERIC_COMPARISON_VALUE_EVENT

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

Table 419. ACI_GAP_NUMERIC_COMPARISON_VALUE_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

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

Table 420. ACI_GAP_KEYPRESS_NOTIFICATION_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)	-

3.3.11 ACI_GAP_PAIRING_REQUEST_EVENT

This event is sent only when SMP mode bit 3 is set to 1. The event is generated in peripheral (a pairing request is received), or in central (when a security request activates a pairing request) case.

The application respond with the ACI_GAP_PAIRING_REQUEST_REPLY command.

Table 421. **ACI_GAP_PAIRING_REQUEST_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection where the event occurred	0x0000 … 0x0EFF
Bonded	1	Indicates if the peer device is bonded	• 0x00: the device is not bonded • 0x01: the device is bonded
Auth_Req	1	Auth_Req field from pairing request (see Bluetooth spec. v.6.0 [Vol 4, Part H, 3.5.1]) or security request (see Bluetooth spec. v.6.0 [Vol 3, Part H, 3.6.7])	-

3.4 ACI GATT/ATT events

In Table 422 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 422. ACI GATT/ATT events list

Command	OpCode	LO	PO	BO	BF	LB
ACI_GATT_ATTRIBUTE_MODIFIED_EVENT	0x0C01	-	Y	-	Y	-
ACI_GATT_PROC_TIMEOUT_EVENT	0x0C02	-	Y	-	Y	-
ACI_ATT_EXCHANGE_MTU_RESP_EVENT	0x0C03	-	Y	-	Y	-
ACI_ATT_FIND_INFO_RESP_EVENT	0x0C04	-	-	-	Y	-
ACI_ATT_FIND_BY_TYPE_VALUE_RESP_EVENT	0x0C05	-	-	-	Y	-
ACI_ATT_READ_BY_TYPE_RESP_EVENT	0x0C06	-	-	-	Y	-
ACI_ATT_READ_RESP_EVENT	0x0C07	-	-	-	Y	-
ACI_ATT_READ_BLOB_RESP_EVENT	0x0C08	-	-	-	Y	-
ACI_ATT_READ_MULTIPLE_RESP_EVENT	0x0C09	-	-	-	Y	-
ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT	0x0C0A	-	-	-	Y	-
ACI_ATT_PREPARE_WRITE_RESP_EVENT	0x0C0C	-	-	-	Y	-
ACI_ATT_EXEC_WRITE_RESP_EVENT	0x0C0D	-	-	-	Y	-
ACI_GATT_INDICATION_EVENT	0x0C0E	-	-	-	Y	-
ACI_GATT_NOTIFICATION_EVENT	0x0C0F	-	-	-	Y	-
ACI_GATT_PROC_COMPLETE_EVENT	0x0C10	-	-	-	Y	-
ACI_GATT_ERROR_RESP_EVENT	0x0C11	-	-	-	Y	-
ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT	0x0C12	-	-	-	Y	-
ACI_GATT_WRITE_PERMIT_REQ_EVENT	0x0C13	-	Y	-	Y	-
ACI_GATT_READ_PERMIT_REQ_EVENT	0x0C14	-	Y	-	Y	-
ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT	0x0C15	-	Y	-	Y	-
ACI_GATT_TX_POOL_AVAILABLE_EVENT	0x0C16	-	Y	-	Y	-
ACI_GATT_SERVER_CONFIRMATION_EVENT	0x0C17	-	Y	-	Y	-
ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT	0x0C18	-	Y	-	Y	-
ACI_GATT_EATT_BEARER_EVENT	0x0C19	-	-	-	-	-
ACI_GATT_MULT_NOTIFICATION_EVENT	0xC1A	-	-	-	-	-
ACI_GATT_NOTIFICATION_COMPLETE_EVENT	0xC1B	-	Y	-	Y	-
ACI_GATT_READ_EXT_EVENT	0xC1D	-	-	-	Y	-
ACI_GATT_INDICATION_EXT_EVENT	0xC1E	-	-	-	Y	-
ACI_GATT_NOTIFICATION_EXT_EVENT	0xC1F	-	-	-	Y	-

3.4.1 ACI_GATT_ATTRIBUTE_MODIFIED_EVENT

This event is generated 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

Table 423. ACI_GATT_ATTRIBUTE_MODIFIED_EVENT parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attr_Handle	2	Handle of the modified attribute	-
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 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	Modified value	-

3.4.2 ACI_GATT_PROC_TIMEOUT_EVENT

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

Table 424. ACI_GATT_PROC_TIMEOUT_EVENT parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)

3.4.3 ACI_ATT_EXCHANGE_MTU_RESP_EVENT

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

Table 425. ACI_ATT_EXCHANGE_MTU_RESP_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

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

Table 426. ACI_ATT_FIND_INFO_RESP_EVENT parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Format	1	Format of Handle_UUID_Pair	-
Event_Data_Length	1	Length of Handle_UUID_Pair in octets	-
Handle_UUID_Pair	Event_Data_Length	A sequence of Handle-UUID pairs: • Format = 1: each pair is [2 octets for handle, 2 octets for UUIDs] • Format = 2: each pair is [2 octets for handle, 16 octets for UUIDs]	-

3.4.5 ACI_ATT_FIND_BY_TYPE_VALUE_RESP_EVENT

This event is generated in response to an ACI_ATT_FIND_BY_TYPE_VALUE_REQ.

Table 427. ACI_ATT_FIND_BY_TYPE_VALUE_RESP_EVENT parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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

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

Table 428. **ACI_ATT_READ_BY_TYPE_RESP_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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

This event is generated in response to a read request. See Section 2.5.24: ACI_GATT_READ_CHAR_VALUE.

Table 429. **ACI_ATT_READ_RESP_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Event_Data_Length	1	Length of following data	-
Attribute_Value	Event_Data_Length	Value of the attribute	-

3.4.8 ACI_ATT_READ_BLOB_RESP_EVENT

This event can be generated during a read long characteristic value procedure, see Section 2.5.26: ACI_GATT_READ_LONG_CHAR_VALUE.

Table 430. **ACI_ATT_READ_BLOB_RESP_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEAw3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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

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

Table 431. **ACI_ATT_READ_MULTIPLE_RESP_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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 attribute handles the request according to the order they were requested.	-

3.4.10 ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT

This event is generated in response to a read by group type request. See Section 2.5.18: ACI_GATT_DISC_ALL_PRIMARY_SERVICES.

Table 432. **ACI_ATT_READ_BY_GROUP_TYPE_RESP_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attribute_Data_Length	1	Size of each attribute data	-
Data_Length	1	Length of Attribute_Data_List, in octets	-
Attribute_Data_List	Data_Length	Attribute data list: • Attribute handle (2 octets) • End group handle (2 octets) • Attribute value (Attribute_Data_Length - 4 octets)	-

3.4.11 ACI_ATT_PREPARE_WRITE_RESP_EVENT

This event is generated in response to an ACI_ATT_PREPARE_WRITE_REQ.

Table 433. **ACI_ATT_PREPARE_WRITE_RESP_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attribute_Handle	2	Handle of the attribute to be written	-
Offset	2	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	Value of the attribute to be written	-

3.4.12 ACI_ATT_EXEC_WRITE_RESP_EVENT

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

Table 434. **ACI_ATT_EXEC_WRITE_RESP_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)

3.4.13 ACI_GATT_INDICATION_EVENT

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

Table 435. **ACI_GATT_INDICATION_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection oriented channel index)
Attribute_Handle	2	Handle of the attribute	-
Attribute_Value_Length	1	Length of Attribute_Value in octets	-
Attribute_Value	Attribute_Value_Length	Current value of the attribute	-

3.4.14 ACI_GATT_NOTIFICATION_EVENT

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

Table 436. **ACI_GATT_NOTIFICATION_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection oriented channel index)
Attribute_Handle	2	Handle of the attribute	-
Attribute_Value_Length	1	Length of Attribute_Value in octets	-
Attribute_Value	Attribute_Value_Length	Current value of the attribute	-

3.4.15 ACI_GATT_PROC_COMPLETE_EVENT

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

Table 437. **ACI_GATT_PROC_COMPLETE_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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

This event is generated when an error response is received from 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.

Table 438. **ACI_GATT_ERROR_RESP_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Req_Opcode	1	The request that generated the error response	-
Attribute_Handle	2	The attribute handle that generated the 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 • 0x12: Database out of synchronization • 0x13: Not allowed value

3.4.17 ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT

This event can be generated during a "Discover Characteristics By UUID" 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.

Table 439. **ACI_GATT_DISC_READ_CHAR_BY_UUID_RESP_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attribute_Handle	2	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

This event is given to the application when a write request, write command, or signed write command is received by the server. This happens only if the event bit for the generation is set when the characteristic was added.

When the event is received, the application must check whether the value requested for write can 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, 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 specified by the application. In case of write/signed write commands, no response is sent to the client, but the attribute is not modified.

Table 440. ACI_GATT_WRITE_PERMIT_REQ_EVENT parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attribute_Handle	2	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

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

Table 441. ACI_GATT_READ_PERMIT_REQ_EVENT parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
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

This event is given to the application when a read multiple request or read by type request is received by the server. This happens only if the event bit for the generation is set when the characteristic was added. On the receptio of this event, the application can update the value of the handles (if needed), and, when done, must send the ACI_GATT_ALLOW_READ command to indicate to the stack that it can send the response to the client.

Table 442. **ACI_GATT_READ_MULTI_PERMIT_REQ_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Number_of_Handles	1	-	-
Handle[i]	2	-	-

3.4.21 ACI_GATT_TX_POOL_AVAILABLE_EVENT

ACI_GATT_TX_POOL_AVAILABLE_EVENT is generated as soon as there is at least one buffer (with a size of ATT_MTU) available in the TX pool each time one of the following GATT commands raises the error code BLE_STATUS_INSUFFICIENT_RESOURCES:

- ACI_GATT_UPDATE_CHAR_VALUE
- ACI_GATT_UPDATE_CHAR_VALUE_EXT
- ACI_GATT_SEND_MULT_NOTIFICATION
- ACI_GATT_WRITE_WITHOUT_RESP
- ACI_GATT_SIGNED_WRITE_WITHOUT_RESP

Table 443. **ACI_GATT_TX_POOL_AVAILABLE_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Not used	-
Available_Buffers	2	Number of available buffers	-

3.4.22 ACI_GATT_SERVER_CONFIRMATION_EVENT

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

Table 444. **ACI_GATT_SERVER_CONFIRMATION_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)

3.4.23 ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT

This event is given to the application when a Prepare write request is received by the server from the client (if the event bit for the generation is set when the characteristic was added). When the event is received, the application must check whether the value requested for write can 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, the value of the attribute is not modified, and an error response is sent to the client, with the error code specified by the application.

Table 445. **ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attribute_Handle	2	Handle of the attribute	-
Offset	2	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_EATT_BEARER_EVENT

This event informs the application of a change in status of the Enhanced ATT bearer handled by the specified L2CAP channel.

Table 446. **ACI_GATT_EATT_BEARER_EVENT parameters**

Parameter	Size	Description	Possible values
Channel_Index	1	Index of the connection-oriented channel to whom the primitive applies	-
EAB_State	1	Enhanced ATT bearer state	• 0x00: enhanced ATT bearer created • 0x01: enhanced ATT bearer terminated
Status	1	Status error code	-

3.4.25 ACI_GATT_MULT_NOTIFICATION_EVENT

This event is generated when a Miultiple_Handle_Value notification is receveid from the server.

Table 447. **ACI_GATT_MULT_NOTIFICATION_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Index of the connection-oriented channel to whom the primitive applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Offset	2	• Bits 14-0: offset in octets from which Attribute_Value data starts • Bit 15: 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 Data, in bytes	-
Data	Data_Length	List of "Handle Length Value" tuples, as defined in Bluetooth Core specification	-

3.4.26 ACI_GATT_NOTIFICATION_COMPLETE_EVENT

This event is generated on the server side after the transmission of all notifications linked with a local update of a characteristic value (if enabled at the creation of the characteristic with GATT_NOTIFY_NOTIFICATION_COMPLETION mask and if the characteristic supports notifications).

Table 448. ACI_GATT_NOTIFICATION_COMPLETE_EVENT parameters

Parameter	Size	Description	Possible values
Attr_Handle	2	Handle of the updated characteristic value	-

3.4.27 ACI_GATT_READ_EXT_EVENT

When 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. To be used when ATT_MTU > (BLE_EVT_MAX_PARAM_LEN - 4), hence ATT_MTU > 251 for the BLE_EVT_MAX_PARAM_LEN default value.

Table 449. ACI_GATT_READ_EXT_EVENT parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Offset	2	Bits 14-0: offset from which Attribute_Value data starts, in octets 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)	-

3.4.28 ACI_GATT_INDICATION_EXT_EVENT

When 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), hence when ATT_MTU > 251 for the BLE_EVT_MAX_PARAM_LEN default value.

Table 450. **ACI_GATT_INDICATION_EXT_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attribute_Handle	2	The handle of the attribute	-
Offset	2	Bits 14-0: offset from which Attribute_Value data starts, in octets 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	-

3.4.29 ACI_GATT_NOTIFICATION_EXT_EVENT

This event is generated instead of ACI_GATT_NOTIFICATION_EVENT when enabled with ACI_GATT_SET_EVENT_MASK, and when a notification is received from the server. It is used instead of ACI_GATT_NOTIFICATION_EVENT when ATT_MTU > (BLE_EVT_MAX_PARAM_LEN - 4), as an example, when ATT_MTU > 251 for the BLE_EVT_MAX_PARAM_LEN default value.

Table 451. **ACI_GATT_NOTIFICATION_EXT_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Specifies the ATT bearer to whom the command applies	• 0x0000 ... 0x0EFF: unenhanced ATT bearer (the parameter is the connection handle) • 0xEA00 ... 0xEA3F: enhanced ATT bearer (the parameter LSB is the connection-oriented channel index)
Attribute_Handle	2	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	Current value of the attribute	-

3.5 ACI L2CAP events

In Table 452 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO / BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 452. ACI L2CAP events commands list

Command	OpCode	LO	PO	BO	BF	LB
ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT	0x0800	-	Y	-	Y	-
ACI_L2CAP_PROC_TIMEOUT_EVENT	0x0801	-	Y	-	Y	-
ACI_L2CAP_CONNECTION_UPDATE_REQ_EVENT	0x0802	-	-	-	Y	-
ACI_L2CAP_COMMAND_REJECT_EVENT	0x080A	-	Y	-	Y	-
ACI_L2CAP_COC_CONNECT_EVENT	0x0810	-	-	-	-	-
ACI_L2CAP_COC_CONNECT_CONFIRM_EVENT	0x0811	-	-	-	-	-
ACI_L2CAP_COC_RECONF_EVENT	0x0812	-	-	-	-	-
ACI_L2CAP_COC_RECONF_CONFIRM_EVENT	0x0813	-	-	-	-	-
ACI_L2CAP_COC_DISCONNECT_EVENT	0x0814	-	-	-	-	-
ACI_L2CAP_COC_FLOW_CONTROL_EVENT	0x0815	-	-	-	-	-
ACI_L2CAP_COC_RX_DATA_EVENT	0x0816	-	-	-	-	-
ACI_L2CAP_COC_TX_POOL_AVAILABLE_EVENT	0x0817	-	-	-	-	-

3.5.1 ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT

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

Table 453. ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT parameters

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

3.5.2 ACI_L2CAP_PROC_TIMEOUT_EVENT

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

Table 454. ACI_L2CAP_PROC_TIMEOUT_EVENT parameters

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection related to this L2CAP procedure	0x0000 … 0x0EFF
Data_Length	1	Length of following data	0
Data	Data_Length	Not used	-

3.5.3 ACI_L2CAP_CONNECTION_UPDATE_REQ_EVENT

The event is given by the L2CAP layer when a connection update request is received from the peripheral. The upper layer receiving this event must respond by sending an ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_RESP command.

Table 455. **ACI_L2CAP_CONNECTION_UPDATE_REQ_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection related to this L2CAP procedure	0x0000 ... 0x0EFF
Identifier	1	Identifier that associates 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. It 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. It 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	Maximum peripheral latency for the connection, in number of connection events	0x0000 ... 0x01F3
Timeout_Multiplier	2	Defines connection timeout parameter as: Timeout Multiplier * 10 ms.	-

3.5.4 ACI_L2CAP_COMMAND_REJECT_EVENT

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

Table 456. **ACI_L2CAP_COMMAND_REJECT_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Connection handle referring to the COS channel where the disconnection has been received	0x0000 … 0x0EFF
Identifier	1	Identifier that associates the request to the response	-
Reason	2	Reason	-
Data_Length	1	Length of following data	-
Data	Data_Length	Data field associated with Reason	-

3.5.5 ACI_L2CAP_COC_CONNECT_EVENT

This event is generated when receiving a valid credit based connection request packet.

Table 457. **ACI_L2CAP_COC_CONNECT_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection where this event occurred	0x0000 ... 0x0EFF
SMPS	2	Simplified Protocol/Service Multiplexer	0x0000 ... 0x00FF
MTU	2	Maximum transmission unit	23 ... 65535
MPS	2	Maximum payload size, in octects	23 ... 248
Initial_Credits	2	Number of K-frames that can be received on the created channel(s) by the L2CAP layer entity sending this packet	0 ... 65535
Channel_Number	1	Number of channels to be created. If set to 0, this parameter requests the creation of one LE credit based connection-oriented channel, otherwise it requests the creation of one or more enhanced credit based connection-oriented channels.	0 ... 5

3.5.6 ACI_L2CAP_COC_CONNECT_CONFIRM_EVENT

This event is generated when receiving a valid credit based connection response packet. See Bluetooth spec. v.5.2 [Vol 3, Part A].

Table 458. **ACI_L2CAP_COC_CONNECT_CONFIRM_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection where this event occurred.	0x0000 ... 0x0EFF
MTU	2	Maximum transmission unit	23 ... 65535
MPS	2	Maximum payload size, in octects	23 ... 248
Initial_Credits	2	Number of K-frames that can be received on the created channel(s) by the L2CAP layer entity sending this packet.	0 ... 65535
Results	2	This parameter indicates the outcome of the request. A value of 0x0000 indicates success, a non-zero value indicates the request is refused.	0x0000 ... 0x000C
Channel_Number	1	Number of created channels. It is the length of Channel_Index_List.	0 ... 5
Channel_Index_List	Channel_Number	List of channel indexes to whom the primitive applies	-

3.5.7 ACI_L2CAP_COC_RECONF_EVENT

This event is generated when receiving a valid credit based reconfigure request packet. See Bluetooth spec. v.5.2 [Vol 3, Part A].

Table 459. **ACI_L2CAP_COC_RECONF_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection where this event occurred	0x0000 ... 0x0EFF
MTU	2	Maximum transmission unit	23 ... 65535
MPS	2	Maximum payload size, in octects	23 ... 248
Channel_Number	1	Number of created channels. It is the length of Channel_Index_List.	1 ... 5
Channel_Index_List	Channel_Number	List of channel indexes to whom the primitive applies	-

3.5.8 ACI_L2CAP_COC_RECONF_CONFIRM_EVENT

This event is generated when receiving a valid credit based reconfigure response packet. See Bluetooth spec. v.5.2 [Vol 3, Part A].

Table 460. **ACI_L2CAP_COC_RECONF_CONFIRM_EVENT parameters**

Parameter	Size	Description	Possible values
Connection_Handle	2	Handle of the connection where this event occurred.	0x0000 ... 0x0EFF
Result	2	This parameter indicates the outcome of the request. A value of 0x0000 indicates success, a non-zero value indicates that the request is refused.	0x0000 ... 0x000C

3.5.9 ACI_L2CAP_COC_DISCONNECT_EVENT

This event is generated when a connection-oriented channel is disconnected following an L2CAP channel termination procedure. See Bluetooth spec. v.5.2 [Vol 3, Part A].

Table 461. **ACI_L2CAP_COC_DISCONNECT_EVENT parameters**

Parameter	Size	Description	Possible values
Channel_Index	1	Index of the connection-oriented channel to whom the primitive applies.	-

3.5.10 ACI_L2CAP_COC_FLOW_CONTROL_EVENT

This event is generated when receiving a valid flow control credit signaling packet. See Bluetooth spec. v.5.2 [Vol 3, Part A].

Table 462. ACI_L2CAP_COC_DISCONNECT_EVENT parameters

Parameter	Size	Description	Possible values
Channel_Index	1	Index of the connection-oriented channel to whom the primitive applies	-
Credits	2	Number of credits the receiving device can increment, corresponding to the number of K-frames that can be sent to the peer device sending the Flow Control Credit packet.	1 … 65535

3.5.11 ACI_L2CAP_COC_RX_DATA_EVENT

This event is generated when receiving a valid K-frame packet on a connection-oriented channel. See Bluetooth spec. v.5.2 [Vol 3, Part A].

Note: For the first K-frame of the SDU, the Information data contains the L2CAP SDU length coded on two octets, followed by the K-frame information payload. For the next K-frames of the SDU the information data only contains the K-frame information payload.

Table 463. ACI_L2CAP_COC_RX_DATA_EVENT parameters

Parameter	Size	Description	Possible values
Channel_Index	1	Index of the connection-oriented channel to whom the primitive applies.	-
Length	2	Length of data in octects	-
Data	Length	Information data	-

3.5.12 ACI_L2CAP_COC_TX_POOL_AVAILABLE_EVENT

Each time ACI_L2CAP_COC_TX_DATA raises the error code BLE_STATUS_INSUFFICIENT_RESOURCES (0x64), the ACI_L2CAP_COC_TX_POOL_AVAILABLE_EVENT event is generated as soon as there is a free buffer available for sending K-frames.

Event parameters: none

3.6 ACI HAL events

In Table 464 "Y" means that the corresponding command applies to the dedicated BLE stack variant, namely LO /BO / PO / BF / LB, respectively, for Link layer only / Beacon only / Peripheral only / Basic feature / Link layer only basic.

Table 464. ACI HAL events commands list

Command	OpCode	LO	PO	BO	BF	LB
ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT	0x0004	Y	Y	Y	Y	Y
ACI_HAL_SCAN_REQ_REPORT_EVENT	0x0005	-	-	-	-	-
ACI_HAL_FW_ERROR_EVENT	0x0006	-	Y	-	Y	-

3.6.1 ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT

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

Table 465. ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT parameters

Parameter	Size	Description	Possible values
Last_State	1	Completed radio events	• 0x00: Idle • 0x01: Advertising • 0x02: Connection event peripheral • 0x03: Scanning • 0x04: Connection request • 0x05: Connection event peripheral • 0x06: TX test mode • 0x07: RX test mode
Next_State	1	Incoming radio events	• 0x00: Idle • 0x01: Advertising • 0x02: Connection event peripheral • 0x03: Scanning • 0x04: Connection request • 0x05: Connection event peripheral • 0x06: TX test mode • 0x07: RX test mode
Next_State_SysTime	4	32-bit absolute current time expressed in internal time units	-
Last_state_Slot	1	Slot number of completed radio event	• 0xFF: Idle • 0x00 … 0x07
Next_State_Slot	1	Slot number of incoming radio event	• 0xFF: Idle • 0x00 … 0x07

3.6.2 ACI_HAL_SCAN_REQ_REPORT_EVENT

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

Table 466. **ACI_HAL_SCAN_REQ_REPORT_EVENT parameters**

Parameter	Size	Description	Possible values
RSSI	1	N Size: 1 octet (signed integer), in dBm	• 127: RSSI not available • -127 ... 20
Peer_Address_Type	1	Peer address type	• 0x00: public device address • 0x01: random device address • 0x02: public identity address • 0x03: random (static) identity address
Peer_Address	6	Public or random device address of the peer device	-

3.6.3 ACI_HAL_FW_ERROR_EVENT

This event is generated to report firmware error information.

Table 467. **ACI_HAL_FW_ERROR_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 • 0x04: COC RX data length too large • 0x05: ECOC already assigned DCID
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. The codes from 0x00 to 0x3E are used for HCI commands (see Core Specification v5.2, Vol. 2, part D), other codes are defined for ACI commands (see the following table).

Table 468. 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
0x11	Unsupported feature or parameter value
0x12	Invalid HCI command parameters
0x13	Remote user terminated connection
0x15	Remote device terminated connection due to power off
0x16	Connection terminated by local host
0x1A	Unsupported remote feature
0x1E	Invalid LL parameters
0x1F	Unspecified error
0x22	LL response timeout
0x23	LL procedure collision
0x24	LMP PDU not allowed
0x28	Instant passed
0x2A	Different transaction collision
0x2E	Channel assessment not supported
0x2F	Insufficient security
0x30	Parameter out of mandatory range
0x38	Host busy - pairing
0x3A	Controller busy
0x3B	Unacceptable connection interval
0x3C	Directed advertising timeout
0x3D	Connection terminated due to MIC failure
0x3E	Connection failed to be established
0x42	Unknown advertising identifier
0x43	Limit reached

Status error code	Description
0x45	Packet too long
0x59	Device in blacklist
0x5A	CSRK not found
0x5B	IRK not found
0x5C	Device not found in DB
0x5E	Device not bonded
0x60	Invalid handle
0x61	Out of handles
0x62	Invalid operation
0x64	Insufficient resources
0x65	Security permission error
0x70	Address not resolved
0x82	No valid slot
0x83	Short window
0x84	New interval failed
0x85	Too large interval
0x86	Slot length failed
0x91	Failed
0x92	Invalid parameters
0x93	Busy
0x95	Pending
0x97	Host error
0x98	Out of memory

5 Tx power level

The following table gives the output power corresponding to the possible values of PA_Level parameter (see Section 2.3.4: ACI_HAL_SET_TX_POWER_LEVEL. These values (at the MCU output) are indicative, and depend upon the PCB layout and the associated components. The inaccuracy can exceed 10 dB in the range -20.85 to -40 dBm.

Table 469. Tx power level

PA_Level	Output power (in dBm)
0x00	-40.0
0x01	-20.85
0x02	-19.75
0x03	-18.85
0x04	-17.60
0x05	-16.50
0x06	-15.25
0x07	-14.10
0x08	-13.15
0x09	-12.05
0x0A	-10.90
0x0B	-9.90
0x0C	-8.85
0x0D	-7.80
0x0E	-6.90
0x0F	-5.90
0x10	-4.95
0x11	-4.00
0x12	-3.15
0x13	-2.45
0x14	-1.80
0x15	-1.30
0x16	-0.85
0x17	-0.50
0x18	-0.15
0x19	0
0x1A	+1.0
0x1B	+2.0
0x1C	+3.0
0x1D	+4.0
0x1E	+5.0
0x1F	+6.0

Revision history

Table 470. **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 392. HCI_HARDWARE_ERROR_EVENT parameters, Table 467. ACI_HAL_FW_ERROR_EVENT parameters
17-Jun-2019	3	Updated: • Table 284. ACI_GATT_UPDATE_CHAR_VALUE input parameters, Table 292. ACI_GATT_SET_EVENT_MASK input parameters, Table 320. ACI_GATT_READ_CHAR_VALUE input parameters, Table 324. ACI_GATT_READ_LONG_CHAR_VALUE input parameters, Table 328. ACI_GATT_WRITE_CHAR_VALUE input parameters, Table 330. ACI_GATT_WRITE_LONG_CHAR_VALUE input parameters, Table 342. ACI_GATT_WRITE_WITHOUT_RESP input parameters, Table 344. ACI_GATT_SIGNED_WRITE_WITHOUT_RESP input parameters, Table 358. ACI_GATT_UPDATE_CHAR_VALUE_EXT input parameters, Table 422. ACI GATT/ATT events list Added: • Section 3.4.28: ACI_GATT_INDICATION_EXT_EVENT and Section 3.4.29: ACI_GATT_NOTIFICATION_EXT_EVENT
25-Sep-2019	4	Updated: • Section 2.4.18: ACI_GAP_CONFIGURE_FILTER_ACCEPT_LIST, Section 2.5.14: ACI_ATT_READ_BY_TYPE_REQ, Section 2.5.25: ACI_GATT_READ_USING_CHAR_UUID, Section 3.4.28: ACI_GATT_INDICATION_EXT_EVENT, Section 4: Status error codes • Table 221. ACI_GAP_TERMINATE_GAP_PROC input parameters, Table 292. ACI_GATT_SET_EVENT_MASK input parameters, Table 389. HCI_DISCONNECTION_COMPLETE_EVENT parameters, Table 417. ACI_GAP_PROC_COMPLETE_EVENT parameters, Table 422. ACI GATT/ATT events list, Table 423. ACI_GATT_ATTRIBUTE_MODIFIED_EVENT parameters, Table 450. ACI_GATT_INDICATION_EXT_EVENT parameters, Table 451. ACI_GATT_NOTIFICATION_EXT_EVENT parameters, Table 468. Status error codes description Added Section 3.4.27: ACI_GATT_READ_EXT_EVENT Removed former Section 4.2 ATT error codes
07-Jan-2020	5	Updated Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters, Table 140. ACI_HAL_READ_CONFIG_DATA input parameters, and Table 414. ACI_GAP_PAIRING_COMPLETE_EVENT parameters. Updated Section 2.5.35: ACI_GATT_WRITE_WITHOUT_RESP and Section 2.5.36: ACI_GATT_SIGNED_WRITE_WITHOUT_RESP.
26-May-2020	6	Updated Table 6. HCI_SET_EVENT_MASK input parameters, Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters, Table 140. ACI_HAL_READ_CONFIG_DATA input parameters, Table 389. HCI_DISCONNECTION_COMPLETE_EVENT parameters, Table 414. ACI_GAP_PAIRING_COMPLETE_EVENT parameters, Table 437. ACI_GATT_PROC_COMPLETE_EVENT parameters, Table 467. ACI_HAL_FW_ERROR_EVENT parameters and Table 468. Status error codes description. Updated title of Section 2.1.34: HCI_LE_READ_REMOTE_FEATURES and Section 3.2.4: HCI_LE_READ_REMOTE_FEATURES_COMPLETE_EVENT. Updated Section 2.4.22: ACI_GAP_START_LIMITED_DISCOVERY_PROC and Section 2.4.23: ACI_GAP_START_GENERAL_DISCOVERY_PROC. Added note in Section 2.4.10: ACI_GAP_INIT. Removed former section HCI_DATA_BUFFER_OVERFLOW_EVENT.
16-Jul-2020	7	Updated Section 3.2.2: HCI_LE_ADVERTISING_REPORT_EVENT and removed former Section 3.6.4 ACI_HAL_DATAPUMP_SENT_EVENT.

Date	Version	Changes
16-Jul-2020	7 (cont'd)	Added three columns in Table 1. HCI commands list, Table 126. HCI testing commands list, Table 136. HAL commands list, Table 169. GAP commands list, Table 274. GATT/ATT commands list, Table 369. L2CAP commands list, Table 388. HCI events commands list, Table 397. HCI LE META events command list, Table 413. ACI GAP events commands list, Table 422. ACI GATT/ATT events list, Table 452. ACI L2CAP events commands list, and Table 464. ACI HAL events commands list.
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 136. HAL commands list and Table 414. ACI_GAP_PAIRING_COMPLETE_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 148. ACI_HAL_GET_LINK_STATUS output parameters. Minor text edits across the whole document.
18-Jun-2021	10	Updated Section 2.1: HCI commands, Section 2.1.6: HCI_SET_CONTROLLER_TO_HOST_FLOW_CONTROL, Section 2.1.7: HCI_HOST_BUFFER_SIZE, 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 3.5: ACI L2CAP events. Updated Table 1. HCI commands list, Table 126. HCI testing commands list, Table 136. HAL commands list, Table 169. GAP commands list, Table 274. GATT/ATT commands list, Table 369. L2CAP commands list, and Table 452. ACI L2CAP events commands list. Added Section 2.1.70: HCI_LE_READ_TRANSMIT_POWER, Section 2.1.73: HCI_LE_SET_PRIVACY_MODE, Section 2.4.44: ACI_GAP_ADDITIONAL_BEACON_START, Section 2.4.45: ACI_GAP_ADDITIONAL_BEACON_STOP, Section 2.4.46: ACI_GAP_ADDITIONAL_BEACON_SET_DATA, Section 3.5.5 to Section 3.5.9, and Section 3.5.5 to Section 3.5.12. Minor text edits across the whole document.
17-Dec-2021	11	Updated Table 1. HCI commands list, Table 126. HCI testing commands list, Table 136. HAL commands list, Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters, Table 169. GAP commands list, Table 171. ACI_GAP_SET_LIMITED_DISCOVERABLE input parameters, Table 173. ACI_GAP_SET_DISCOVERABLE input parameters, Table 175. ACI_GAP_SET_DIRECT_CONNECTABLE input parameters, Table 187. ACI_GAP_INIT input parameters, Table 215. ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC input parameters, Table 217. ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC input parameters, Table 219. ACI_GAP_CREATE_CONNECTION input parameters, Table 229. ACI_GAP_SET_BROADCAST_MODE input parameters, Table 231. ACI_GAP_START_OBSERVATION_PROC input parameters, Table 241. ACI_GAP_GET_OOB_DATA output parameters, Table 242. ACI_GAP_SET_OOB_DATA input parameters, Table 244. ACI_GAP_ADD_DEVICES_TO_RESOLVING_LIST input parameters, Table 397. HCI LE META events command list, Table 422. ACI GATT/ATT events list, Table 467. ACI_HAL_FW_ERROR_EVENT parameters, and Table 468. Status error codes description.

Date	Version	Changes
17-Dec-2021	11 (cont'd)	Added Section 2.1.58: HCI_LE_SET_ADVERTISING_SET_RANDOM_ADDRESS, Section 2.1.59: HCI_LE_SET_EXTENDED_ADVERTISING_PARAMETERS, Section 2.1.60: HCI_LE_SET_EXTENDED_ADVERTISING_DATA, Section 2.1.61: HCI_LE_SET_EXTENDED_SCAN_RESPONSE_DATA, Section 2.1.62: HCI_LE_SET_EXTENDED_ADVERTISING_ENABLE, Section 2.1.63: HCI_LE_READ_MAXIMUM_ADVERTISING_DATA_LENGTH, Section 2.1.64: HCI_LE_READ_NUMBER_OF_SUPPORTED_ADVERTISING_SETS, Section 2.1.65: HCI_LE_REMOVE_ADVERTISING_SET, Section 2.1.66: HCI_LE_CLEAR_ADVERTISING_SETS, Section 2.1.67: HCI_LE_SET_EXTENDED_SCAN_PARAMETERS, Section 2.1.68: HCI_LE_SET_EXTENDED_SCAN_ENABLE, Section 2.1.69: HCI_LE_EXTENDED_CREATE_CONNECTION, Section 2.3.13: ACI_HAL_SET_PERIPHERAL_LATENCY, Section 2.4.42: ACI_GAP_ADD_DEVICES_TO_LIST, Section 2.4.47: ACI_GAP_ADV_SET_CONFIGURATION, Section 2.4.48: ACI_GAP_ADV_SET_ENABLE, Section 2.4.49: ACI_GAP_ADV_SET_ADV_DATA, Section 2.4.50: ACI_GAP_ADV_SET_SCAN_RESP_DATA, Section 2.4.51: ACI_GAP_ADV_REMOVE_SET, Section 2.4.52: ACI_GAP_ADV_CLEAR_SETS, Section 2.4.53: ACI_GAP_ADV_SET_RANDOM_ADDRESS, Section 3.2.12: HCI_LE_EXTENDED_ADVERTISING_REPORT_EVENT, Section 3.2.13: HCI_LE_SCAN_TIMEOUT_EVENT, Section 3.2.14: HCI_LE_ADVERTISING_SET_TERMINATED_EVENT, Section 3.2.15: HCI_LE_SCAN_REQUEST_RECEIVED_EVENT, and Section 3.2.16: HCI_LE_CHANNEL_SELECTION_ALGORITHM_EVENT. Updated Section 2.1.19: HCI_LE_READ_ADVERTISING_PHYSICAL_CHANNEL_TX_POWER, Section 2.1.37: HCI_LE_ENABLE_ENCRYPTION, Section 2.1.39: HCI_LE_LONG_TERM_KEY_REQUEST_NEGATIVE_REPLY, Section 2.2.4: HCI_LE_RECEIVER_TEST_V2, Section 2.2.5: HCI_LE_TRANSMITTER_TEST_V2, Section 2.3.2: ACI_HAL_WRITE_CONFIG_DATA, and Section 2.4.4: ACI_GAP_SET_DIRECT_CONNECTABLE. Removed former Section 2.4.24 ACI_GAP_START_NAME_DISCOVERY_PROC. Minor text edits across the whole document.
29-Apr-2022	12	Updated Table 1. HCI commands list, Table 22. HCI_LE_SET_EVENT_MASK input parameters, Table 129. HCI_LE_TRANSMITTER_TEST input parameters, Table 136. HAL commands list, Table 169. GAP commands list, and Table 397. HCI LE META events command list. Added Section 2.1.71: HCI_LE_READ_RF_PATH_COMPENSATION and Section 2.1.72: HCI_LE_WRITE_RF_PATH_COMPENSATION.
17-Jun-2022	13	Updated Introduction, Section 2.1: HCI commands, Section 2.2: HCI testing commands, Section 2.3: HAL commands, Section 2.4: GAP commands, Section 2.4.10: ACI_GAP_INIT, Section 2.5: GATT/ATT commands, Section 2.6: L2CAP commands, Section 3.1: HCI events, Section 3.2: HCI LE META events, Section 3.3: ACI GAP events, Section 3.4: ACI GATT/ATT events, Section 3.5: ACI L2CAP events, and Section 3.6: ACI HAL events. Updated Table 1. HCI commands list, Table 116. HCI_LE_EXTENDED_CREATE_CONNECTION input parameters, Table 126. HCI testing commands list, Table 136. HAL commands list, Table 169. GAP commands list, Table 274. GATT/ATT commands list, Table 369. L2CAP commands list, Table 388. HCI events commands list, Table 397. HCI LE META events command list, Table 413. ACI GAP events commands list, Table 422. ACI GATT/ATT events list, Table 452. ACI L2CAP events commands list, and Table 464. ACI HAL events commands list. Minor text edits across the whole document.
06-Jul-2022	14	Updated Section 2.5.6: ACI_GATT_UPDATE_CHAR_VALUE, and Section 3.2.10: HCI_LE_DIRECTED_ADVERTISING_REPORT_EVENT. Added Section 2.5.46: ACI_GATT_STORE_DB. Minor text edits across the whole document.

Date	Version	Changes
06-Jul-2022	14 (cont'd)	Updated Table 116. HCI_LE_EXTENDED_CREATE_CONNECTION input parameters, Table 136. HAL commands list, Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters, Table 148. ACI_HAL_GET_LINK_STATUS output parameters. Table 252. ACI_GAP_ADDITIONAL_BEACON_START input parameters, Table 263. ACI_GAP_ADV_SET_SCAN_RESP_DATA input parameters, Table 264. ACI_GAP_ADV_SET_SCAN_RESP_DATA output parameters, Table 267. ACI_GAP_ADV_CLEAR_SETS output parameters, Table 274. GATT/ATT commands list, Table 407. HCI_LE_DIRECTED_ADVERTISING_REPORT_EVENT parameters, and Table 465. ACI_HAL_END_OF_RADIO_ACTIVITY_EVENT parameters.
21-Jul-2022	15	Updated Table 164. ACI_HAL_READ_RAW_RSSI output parameters.
16-Aug-2022	16	Updated Table 6. HCI_SET_EVENT_MASK input parameters, Table 22. HCI_LE_SET_EVENT_MASK input parameters, Table 97. HCI_LE_SET_ADVERTISING_SET_RANDOM_ADDRESS input parameters, Table 112. HCI_LE_SET_EXTENDED_SCAN_PARAMETERS input parameters, Table 116. HCI_LE_EXTENDED_CREATE_CONNECTION input parameters, Table 119. HCI_LE_READ_RF_PATH_COMPENSATION output parameters, Table 120. HCI_LE_WRITE_RF_PATH_COMPENSATION input parameters, Table 136. HAL commands list, Table 211. ACI_GAP_START_GENERAL_DISCOVERY_PROC input parameters, Table 213. ACI_GAP_START_AUTO_CONNECTION_ESTABLISH_PROC input parameters, Table 215. ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC input parameters, Table 217. ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC input parameters, Table 219. ACI_GAP_CREATE_CONNECTION input parameters, and Table 231. ACI_GAP_START_OBSERVATION_PROC input parameters. Added Section 2.3.14: ACI_HAL_READ_RSSI. Minor text edits across the whole document.
04-Jan-2023	17	Updated Section 2.1.4: HCI_RESET, Section 2.3.5: ACI_HAL_LE_TX_TEST_PACKET_NUMBER, Section 2.3.13: ACI_HAL_SET_PERIPHERAL_LATENCY, Section 2.3.14: , Section 2.4.1: ACI_GAP_SET_NON_DISCOVERABLE, Section 2.4.3: ACI_GAP_SET_DISCOVERABLE, Section 2.4.4: ACI_GAP_SET_DIRECT_CONNECTABLE, Section 2.4.12: ACI_GAP_SET_UNDIRECTED_CONNECTABLE, Section 2.4.32: ACI_GAP_SET_BROADCAST_MODE, Section 2.5.2: ACI_GATT_ADD_SERVICE, and Section 2.5.4: ACI_GATT_ADD_CHAR. Added Section 2.5.47: ACI_GATT_SET_MULT_NOTIFICATION, Section 2.5.48: ACI_GATT_READ_MULTIPLE_VAR_CHAR_VALUE, Section 3.4.24: ACI_GATT_EATT_BEARER_EVENT, Section 3.4.25: ACI_GATT_MULT_NOTIFICATION_EVENT, and Section 5: Tx power level. Minor text edits across the whole document.

Date	Version	Changes
04-Jan-2023	17 (cont'd)	Updated Table 13. HCI_HOST_BUFFER_SIZE input parameters, Table 119. HCI_LE_READ_RF_PATH_COMPENSATION output parameters, Table 136. HAL commands list, Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters, Table 145. ACI_HAL_TONE_START input parameters, Table 148. ACI_HAL_GET_LINK_STATUS output parameters, Table 165. ACI_HAL_RX_START input parameters, Table 274. GATT/ATT commands list, Table 276. ACI_GATT_ADD_SERVICE input parameters, Table 280. ACI_GATT_ADD_CHAR input parameters, Table 308. ACI_GATT_DISC_ALL_PRIMARY_SERVICES input parameters, Table 310. ACI_GATT_DISC_PRIMARY_SERVICE_BY_UUID input parameters, Table 312. ACI_GATT_FIND_INCLUDED_SERVICES input parameters, Table 316. ACI_GATT_DISC_CHAR_BY_UUID input parameters, Table 318. ACI_GATT_DISC_ALL_CHAR_DESC input parameters, Table 320. ACI_GATT_READ_CHAR_VALUE input parameters, Table 322. ACI_GATT_READ_USING_CHAR_UUID input parameters, Table 324. ACI_GATT_READ_LONG_CHAR_VALUE input parameters, Table 326. ACI_GATT_READ_MULTIPLE_CHAR_VALUE input parameters, Table 328. ACI_GATT_WRITE_CHAR_VALUE input parameters, Table 330. ACI_GATT_WRITE_LONG_CHAR_VALUE input parameters, Table 332. ACI_GATT_WRITE_CHAR_RELIABLE input parameters, Table 334. ACI_GATT_WRITE_LONG_CHAR_DESC input parameters, Table 336. ACI_GATT_READ_LONG_CHAR_DESC input parameters, Table 338. ACI_GATT_WRITE_CHAR_DESC input parameters, Table 340. ACI_GATT_READ_CHAR_DESC input parameters, Table 342. ACI_GATT_WRITE_WITHOUT_RESP input parameters, Table 344. ACI_GATT_SIGNED_WRITE_WITHOUT_RESP input parameters, Table 346. ACI_GATT_CONFIRM_INDICATION input parameters, Table 348. ACI_GATT_WRITE_RESP input parameters, Table 350. ACI_GATT_ALLOW_READ input parameters, Table 360. ACI_GATT_DENY_READ input parameters, Table 422. ACI GATT/ATT events list, Table 423. ACI_GATT_ATTRIBUTE_MODIFIED_EVENT parameters, Table 424. ACI_GATT_PROC_TIMEOUT_EVENT parameters, Table 426 to Table 434, Table 437 to Table 443, Table 445. ACI_GATT_PREPARE_WRITE_PERMIT_REQ_EVENT parameters, Table 451. ACI_GATT_NOTIFICATION_EXT_EVENT parameters, and Table 453. ACI_L2CAP_CONNECTION_UPDATE_RESP_EVENT parameters.
19-Jan-2023	18	Updated Table 358. ACI_GATT_UPDATE_CHAR_VALUE_EXT input parameters, Table 392. HCI_HARDWARE_ERROR_EVENT parameters, Table 435. ACI_GATT_INDICATION_EVENT parameters, Table 436. ACI_GATT_NOTIFICATION_EVENT parameters, Table 438. ACI_GATT_ERROR_RESP_EVENT parameters, Table 444. ACI_GATT_SERVER_CONFIRMATION_EVENT parameters, Table 449. ACI_GATT_READ_EXT_EVENT parameters, and Table 450. ACI_GATT_INDICATION_EXT_EVENT parameters. Updated Section 3.3.8: ACI_GAP_ADDR_NOT_RESOLVED_EVENT.
23-Feb-2023	19	Updated Table 1. HCI commands list, Table 397. HCI LE META events command list, and Table 468. Status error codes description. Updated Section 2.1.22: HCI_LE_SET_ADVERTISING_ENABLE and Section 3.1.1: HCI_DISCONNECTION_COMPLETE_EVENT. Minor text edits across the whole document.
12-May-2023	20	Updated Section Introduction, Section 2.1.1: HCI_DISCONNECT to Section 2.1.73: HCI_LE_SET_PRIVACY_MODE, Section 2.2.1: HCI_LE_RECEIVER_TEST to Section 2.2.5: HCI_LE_TRANSMITTER_TEST_V2, Section 2.3.11: ACI_HAL_SET_EVENT_MASK, Section 2.3.13: ACI_HAL_SET_PERIPHERAL_LATENCY, Section 2.4.4: ACI_GAP_SET_DIRECT_CONNECTABLE, Section 2.4.13: ACI_GAP_PERIPHERAL_SECURITY_REQ, Section 2.4.17: ACI_GAP_SET_EVENT_MASK, Section 2.4.18: ACI_GAP_CONFIGURE_FILTER_ACCEPT_LIST, Section 2.5.4: ACI_GATT_ADD_CHAR, Section 2.5.5: ACI_GATT_ADD_CHAR_DESC, Section 2.5.6: ACI_GATT_UPDATE_CHAR_VALUE, Section 2.5.10: ACI_GATT_SET_EVENT_MASK, Section 2.5.43: ACI_GATT_UPDATE_CHAR_VALUE_EXT, Section 3.1.1: HCI_DISCONNECTION_COMPLETE_EVENT, Section 3.3.5: ACI_GAP_PERIPHERAL_SECURITY_INITIATED_EVENT, and Section 3.1.6: HCI_ENCRYPTION_KEY_REFRESH_COMPLETE_EVENT.

Date	Version	Changes
12-May-2023	20 (cont'd)	Added Section 2.1.74: HCI_LE_GENERATE_DHKEY_V2 and Section 3.4.26: ACI_GATT_NOTIFICATION_COMPLETE_EVENT. Changed "0xEA00 ... 0xEA1F" with "0xEA00 ... 0xEA3F" as range of possible values for Connection_Handle parameter throughout the whole document. Updated Table 1. HCI commands list, Table 136. HAL commands list, Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters, Table 169. GAP commands list, Table 201. ACI_GAP_SET_EVENT_MASK input parameters, Table 240. ACI_GAP_GET_OOB_DATA input parameters, Table 241. ACI_GAP_GET_OOB_DATA output parameters, Table 242. ACI_GAP_SET_OOB_DATA input parameters, Table 413. ACI GAP events commands list, and Table 422. ACI GATT/ATT events list. Minor text edits across the whole document.
14-Jul-2023	21	Updated Table 41. HCI_LE_CREATE_CONNECTION input parameters, Table 50. HCI_LE_CONNECTION_UPDATE input parameters, Table 95. HCI_LE_SET_PHY input parameters, Table 116. HCI_LE_EXTENDED_CREATE_CONNECTION input parameters, Table 142. ACI_HAL_SET_TX_POWER_LEVEL input parameters, Table 370. ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_REQ input parameters, Table 372. ACI_L2CAP_CONNECTION_PARAMETER_UPDATE_RESP input parameters, Table 374. ACI_L2CAP_COC_CONNECT input parameters, Table 376. ACI_L2CAP_COC_CONNECT_CONFIRM input parameters, Table 378. ACI_L2CAP_COC_RECONF input parameters, Table 457. ACI_L2CAP_COC_CONNECT_EVENT parameters, Table 458. ACI_L2CAP_COC_CONNECT_CONFIRM_EVENT parameters, and Table 459. ACI_L2CAP_COC_RECONF_EVENT parameters. Updated list of generated events in Section 2.1.25: HCI_LE_CREATE_CONNECTION, Section 2.1.26: HCI_LE_CREATE_CONNECTION_CANCEL, Section 2.1.62: HCI_LE_SET_EXTENDED_ADVERTISING_ENABLE, Section 2.1.69: HCI_LE_EXTENDED_CREATE_CONNECTION, Section 2.4.26: ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC, and Section 2.4.27: ACI_GAP_CREATE_CONNECTION. Updated Section 2.3.4: ACI_HAL_SET_TX_POWER_LEVEL, Section 2.4.20: ACI_GAP_CLEAR_SECURITY_DB, Section 2.4.27: ACI_GAP_CREATE_CONNECTION, and Section 2.5.6: ACI_GATT_UPDATE_CHAR_VALUE. Minor text edits across the whole document.
03-Oct-2023	22	Updated Section 2.4.47: ACI_GAP_ADV_SET_CONFIGURATION. Updated Table 99. HCI_LE_SET_EXTENDED_ADVERTISING_PARAMETERS input parameters, Table 112. HCI_LE_SET_EXTENDED_SCAN_PARAMETERS input parameters, Table 132. HCI_LE_RECEIVER_TEST_V2 input parameters, and Table 257. ACI_GAP_ADV_SET_CONFIGURATION input parameters.
10-Nov-2023	23	Updated Table 16. HCI_READ_LOCAL_VERSION_INFORMATION output parameters, Table 93. HCI_LE_SET_DEFAULT_PHY input parameters, Table 95. HCI_LE_SET_PHY input parameters, Table 169. GAP commands list, Table 280. ACI_GATT_ADD_CHAR input parameters, Table 282. ACI_GATT_ADD_CHAR_DESC input parameters, Table 467. ACI_HAL_FW_ERROR_EVENT parameters, and Table 468. Status error codes description. Added Section 2.4.54: ACI_GAP_EXT_START_SCAN and Section 2.4.55: ACI_GAP_EXT_CREATE_CONNECTION. Updated Section 2.4.35: ACI_GAP_IS_DEVICE_BONDED and Section 2.4.41: ACI_GAP_REMOVE_BONDED_DEVICE.
12-Dec-2023	24	Updated Section 2.3.2: ACI_HAL_WRITE_CONFIG_DATA, Section 2.5.6: ACI_GATT_UPDATE_CHAR_VALUE, and Section 3.4.21: ACI_GATT_TX_POOL_AVAILABLE_EVENT. Updated Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters.
28-Feb-2024	25	Updated Table 252. ACI_GAP_ADDITIONAL_BEACON_START input parameters. Updated Section 5: Tx power level.

Date	Version	Changes
05-Jun-2024	26	Updated Table 1. HCI commands list, Table 136. HAL commands list, Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters,Table 284. ACI_GATT_UPDATE_CHAR_VALUE input parameters, Table 328. ACI_GATT_WRITE_CHAR_VALUE input parameters, Table 332. ACI_GATT_WRITE_CHAR_RELIABLE input parameters, Table 334. ACI_GATT_WRITE_LONG_CHAR_DESC input parameters, Table 338. ACI_GATT_WRITE_CHAR_DESC input parameters, Table 358. ACI_GATT_UPDATE_CHAR_VALUE_EXT input parameters, Table 414. ACI_GAP_PAIRING_COMPLETE_EVENT parameters, Table 454. ACI_L2CAP_PROC_TIMEOUT_EVENT parameters, Table 455. ACI_L2CAP_CONNECTION_UPDATE_REQ_EVENT parameters, and Table 456. ACI_L2CAP_COMMAND_REJECT_EVENT parameters. Added Section 2.3.15: ACI_HAL_EAD_ENCRYPT_DECRYPT. Updated Section 2.4.22: ACI_GAP_START_LIMITED_DISCOVERY_PROC, Section 2.4.23: ACI_GAP_START_GENERAL_DISCOVERY_PROC, Section 2.4.25: ACI_GAP_START_GENERAL_CONNECTION_ESTABLISH_PROC, Section 2.4.26: ACI_GAP_START_SELECTIVE_CONNECTION_ESTABLISH_PROC, and Section 2.4.33: ACI_GAP_START_OBSERVATION_PROC. Minor text edits across the whole document.
24-Sep-2024	27	Updated reference Bluetooth standard, from 5.4 to 6.0. Updated document title, Section 2.4.14: ACI_GAP_UPDATE_ADV_DATA, and Section 2.4.35: ACI_GAP_IS_DEVICE_BONDED. Updated Table 1. HCI commands list, Table 132. HCI_LE_RECEIVER_TEST_V2 input parameters, Table 138. ACI_HAL_WRITE_CONFIG_DATA input parameters, Table 169. GAP commands list, Table 234. ACI_GAP_IS_DEVICE_BONDED input parameters, and Table 413. ACI GAP events commands list. Added Section 2.4.43: ACI_GAP_PAIRING_REQUEST_REPLY and Section 3.3.11: ACI_GAP_PAIRING_REQUEST_EVENT. Minor text edits across the whole document.

Contents

List of tables